2006

Unified data mining using stable patterns

Pranali Khadpe
San Jose State University

UNIFIED DATA MINING USING STABLE PATTERNS

A Thesis

Presented To

The Faculty of the Department of Computer Engineering

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Masters of Science in Computer Engineering

by

Pranali Khadpe

May 2006

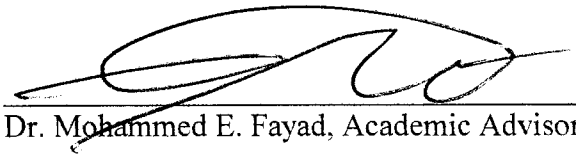
INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

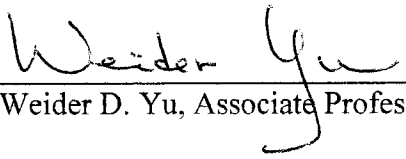
In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

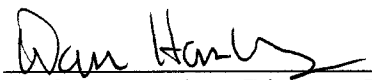
APPROVED FOR THE DEPARTMENT OF COMPUTER ENGINEERING



Dr. Mohammed E. Fayad, Academic Advisor

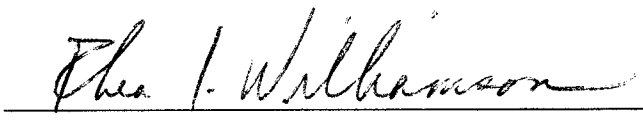


Dr. Weider D. Yu, Associate Professor Computer Engineering Department



Prof. Dan Harkey, Director, Enterprise Software Technologies Program

APPROVED FOR THE UNIVERSITY



ABSTRACT

UNIFIED DATA MINING USING STABLE PATTERNS

by Pranali B. Khadpe

This thesis examines the current data mining tools and builds a new tool using various data mining techniques. Research on this subject reveals that the current data mining tools implement any one of the available techniques. So the user or company has to either use many tools to get the optimum result, or be satisfied with the result obtained by the selected tool. Building a tool, which would give the user the option of using all the available data mining algorithms, is the motivation behind this thesis.

For this purpose, the theory of Software Stability Model is exploited to implement a pattern language. This pattern language is the bridge, which will lead us to the implementation of the tool. Using Software Stability paradigm, some Stable Analysis Patterns and Stable Design Patterns are developed, which can be applied across diverse domains and can be extended in any applications of interest.

Table of Contents

CHAPTER 1 Introduction 1
1.1 Introduction to Data Mining 2
1.2 Representation of the Problem 3
1.3 Software Stability Model Overview 5
1.4 Research Methodology 7
1.5 Conducting the Research Methodology 10
1.6 Thesis Layout 11
CHAPTER 2 Software Stability Model and Stable Pattern Language 12
2.1 Pattern Language Evolution - Stability Concepts and Pattern Languages 12
2.2 Description of Existing Tools 15
2.2.1 CART 15
2.2.2 XPertRule 18
2.3 Unified Data Mining Solutions 21
2.4 Comparative Analysis of Tools 23
2.4.1 Reusability 24
2.4.2 Applicability 25
2.4.3 Adaptive Features 25
2.4.4 Core Knowledge 26
2.4.5 Stability 27
2.4.6 Support for Processes 27
2.4.7 Support for Algorithms 28
2.4.8 Level of Exploration 28
2.4.9 White Papers and Resources 29
2.4.10 Level of Support for Data Mining Techniques 30
2.4.11 Overview of Comparative Analysis 31
CHAPTER 3 Data Mining Goals 35
3.1 Overview of Goals 36
3.1.1 Discovery 36
3.1.2 Knowledge 38
3.1.3 Analysis 41
3.2 Subgoals 44
3.2.1 Subgoals of Discovery 44
3.2.2 Subgoals of Knowledge 45
3.2.3 Subgoals of Analysis 46
CHAPTER 4 Data Mining Capabilities 47
4.1 Overview of Data Mining Capabilities 48
4.1.1 Any Data Collection 48
4.1.2 Any Data Mining 51
4.1.3 Any Data Preparation 53
4.1.4 Any Data Mining Mechanism 53
4.1.5 Any Data Selection 53

4.1.6 Capabilities Related to Goals 54
CHAPTER 5 Knowledge Map and Stable Architecture Patterns 55
5.1 Development of KM Through Scenarios 55
5.1.1 KM Through Goal Scenarios 56
5.1.2 KM Through Capabilities 59
5.2 Data Mining Goals Realized Through the Capabilities 62
5.3 Data Mining KM 63
5.4 Architecture Assessment 71
5.4.1 Properties of KM 71
CHAPTER 6 Data Mining Development Scenarios and Implementation Details 75
6.1 Type Oriented Paradigm 77
6.1.1 Inheritance 77
6.1.2 Containment 80
6.2 Hook 82
6.2.1 Inheritance Hook 84
6.2.2 Aggregation Hook 85
6.3 Model Based Architecture 86
6.3.1 Data Mining Model Based Architecture 87
CHAPTER 7 Data Mining Deployment 95
7.1 Deployment of Data Mining Patterns in Different Application Domains 95
7.1.1 Quality Factors 97
7.2 Validations and Verification 100
CHAPTER 8 Conclusions 102
8.1 Challenges 102
8.2 Future Work 103
8.3 Conclusions 105
REFERENES 107
APPENDIX A Stable Analysis Pattern - Discovery Pattern 113
A.1 Pattern Documentation 113
A.2 Applicability: Case Study 1 - Discovery of Vitamin K 119
A.2.1 Use Case Description 121
A.2.2 Behavior Diagram 123
A.3 Applicability: Case Study 2 - Research Application 125
A.3.1 Use Case Description 126
A.3.2 Behavior Diagram 129
A.4 Applicability: Case Study 3 - Planetary Research 130
A.4.1 Use Case Description 131
A.4.2 Behavior Diagram 134
APPENDIX B Stable Design Pattern - Any Data Mining Pattern 135
B.1 Pattern Documentation 135
B.2 Applicability: Case Study 1 - Moviegoer's Application 141
B.2.1 Use Case Description 141
B.2.2 Behavior Diagram 146
B.3 Applicability: Case Study 2 - Credit Card Fraud Detection Application 147

B.3.1 Use Case Description 147
B.3.2 Behavior Diagram 151
APPENDIX C Implementation and Code 152
C.1 Implementation Details 152
C.1.1 Discovery Stable Analysis Pattern 152
C.1.2 Unique Segment of Code 154
APPENDIX D Instructions and Demo 172
D.1 Setup Requirements 172
D.1.1 Installing the Java Development Kit (JDK) 172
D.1.2 Configuring Tomcat 172
D.1.3 Deploying the Application 173
D.1.4 Target System 174
D.2 Demo Instructions and Snapshots 174
D.2.1 Discovery Stable Analysis Pattern - Knowledge Discovery Application 174
D.2.2 Discovery Stable Analysis Pattern - Planet Discovery Application 183
D.2.3 Any Data Mining Pattern - Credit Card Fraud Detection Application 190
D.2.4 Any Data Mining Pattern - Moviegoer's Application 193

List of Figures

Figure 1-1 Scope of Applications Using the SSM 6
Figure 1-2 Stable Pattern Language - Testing 8
Figure 2-1 CART's Decision Tree Mechanism [23] 16
Figure 2-2 CART's Tree Navigator [23] 17
Figure 2-3 CART's Navigator Reports [23]. 17
Figure 2-4 XPertRule's Tree Induction [24] 19
Figure 2-5 XPertRule's Interactive Induction [24] 20
Figure 2-6 XPertRule's Pattern Exploration and Validation Tool [24] 21
Figure 3-1 Discovery Pattern (taken from Dr. Fayad's patterns archive) 37
Figure 3-2 Knowledge Pattern (taken from Dr. Fayad's patterns archive) 39
Figure 3-3 Analysis Pattern (taken from Dr. Fayad's patterns archive) 42
Figure 4-1 Any Data Collection Pattern (taken from Dr. Fayad's patterns archive) 49
Figure 4-2 Any Data Mining Pattern (taken from Dr. Fayad's patterns archive). 52
Figure 5-1 Discovery Pattern (taken from Dr. Fayad's patterns archive) 59
Figure 5-2 Any Collection Pattern (taken from Dr. Fayad's patterns archive) 62
Figure 5-3 Overview of KM 64
Figure 5-4 Data Mining KM 65
Figure 5-5 Partition I 66
Figure 5-6 Partition II 67
Figure 5-7 Data Mining KM with Other KMs. 68
Figure 5-8 Partition III 69
Figure 5-9 Data Mining KM with External and Remote KMs. 71
Figure 5-10 Example of Architecture Generated Using Discovery and Analysis 72
Figure 6-1 Representation of Type, Class, and Interfaces in Inheritance 78
Figure 6-2 Representation of Type, Class, and Interfaces in Collection 79
Figure 6-3 Example 1 Representation of Type, Class and Interfaces 81
Figure 6-4 Representation of Type, Class and Interfaces - Collection 82
Figure 6-5 Model Driven Architecture [20] 86
Figure 6-6 Data Mining Model Based Architecture 87
Figure 6-7 Model Based Architecture in Credit Card Fraud Detection Application 91
Figure 6-8 Model Based Architecture in Knowledge Discovery Application. 93
Figure 7-1 Overview of Discovery Pattern and its Applications 96
Figure 7-2 Overview of Data Mining Pattern and its Applications 96
Figure 7-3 KM with Quality Factors 99
Figure A-1 Discovery Stable Analysis Pattern 116
Figure A-2 Class Diagram for Case Study 1 120
Figure A-3 Sequence Diagram for Discover Vitamin K Use Case. 124
Figure A-4 Knowledge Discovery Application 125
Figure A-5 Knowledge Discovery Sequence Diagram 129
Figure A-6 Planetary System Application 130
Figure A-7 Sequence Diagram for Discovery of a Planet 134

Figure B-1 Any Data Mining Pattern (taken from Dr. Fayad's Pattern archive)..... 138
Figure B-2 Any Data Mining Pattern in Moviegoer's Application 145
Figure B-3 Sequence Diagram for Application 1..... 146
Figure B-4 Credit Card Fraud Detection Application Using Any Data Mining Pattern. 150
Figure B-5 Sequence Diagram for Application 151
Figure C-1 Discovery Stable Analysis Pattern..... 152
Figure C-2 Discovery Stable Analysis Pattern in Knowledge Discovery Application... 153
Figure C-3 Second Level Pattern for Any Discovery Pattern..... 156
Figure C-4 Second Level Pattern for Any Type 160
Figure C-5 Second Level Pattern for Any Discovery Mechanism or Any Service. 163
Figure C-6 Second Level Pattern for Any Evidence 167
Figure C-7 Second Level Pattern for Any Actor Pattern..... 170
Figure D-1 Login.html..... 176
Figure D-2 Knowledge Discovery LoginServlet 177
Figure D-3 Knowledge Discovery.jsp 178
Figure D-4 Information About Algorithms - Clustering 179
Figure D-5 Information About Algorithms - Text Mining 180
Figure D-6 Implementation of Clustering Algorithm..... 181
Figure D-7 Implementation of Text Mining Algorithm 182
Figure D-8 Pattern Servlet to Record the Pattern..... 183
Figure D-9 Planetary System 185
Figure D-10 Observation Servlet 186
Figure D-11 List of Recorded Observations 187
Figure D-12 Planet Recording Screen 188
Figure D-13 List of Recorded Planets 189
Figure D-14 Fdetection.jsp 191
Figure D-15 Unsupervised Learning (usl.jsp)..... 192
Figure D-16 ClusteredInfo.jsp 192
Figure D-17 ClusteredDetail.jsp 193
Figure D-18 Moviegoer's Application 195
Figure D-19 Prediction.jsp..... 195
Figure D-20 Estimation.jsp..... 196
Figure D-21 EstimationSurvey.jsp 197
Figure D-22 EstimationPerRating.jsp..... 197
Figure D-23 EstimationPeoplePerMovie.jsp 198

List of Tables

Table 2-1 Research Methodology Overview 13
Table 2-2 Overview of Comparative Analysis of Data Mining Tools 31
Table 2-3 Comparison of Tools for Algorithms 32
Table 2-4 Comparison of Tools for Processes 33
Table 2-5 Comparison of Tools for Exploration Process 34
Table 2-6 Comparison of Tools for Data Mining Techniques 34
Table 4-1 List of the Rest Capabilities 54
Table 5-1 Discovery - Scenario 1 57
Table 5-2 Discovery - Scenario 2 58
Table 5-3 Discovery - Final Model 58
Table 5-4 Any Collection - Scenario 1 60
Table 5-5 Any Collection - Scenario 2 61
Table 5-6 Any Collection - Final Model 61
Table 5-7 Goals 63
Table 6-1 Specification of EBT Discovery 75
Table 6-2 Specification of EBT Knowledge 75
Table 6-3 Specification of EBT Analysis 76
Table 6-4 Specification of BO Any Data Preparation 76
Table 6-5 Specification of BO Any Data Collection 76
Table 6-6 Example 1 of Type Names an Interface 77
Table 6-7 Example 1 of Class Implements a Type 78
Table 6-8 Type Names an Interface – Collection 78
Table 6-9 Class Iimplements a Type – Collection 79
Table 6-10 Example 2 of Type Names an Interface 80
Table 6-11 Example 2 of Class Implements a Type Any Media 80
Table 6-12 Type Names an Interface - Mechanism 81
Table 6-13 Class Implements a Type - Mechanism 82
Table 6-14 Example of Hooks Used in Knowledge Discovery Application 83
Table 6-15 Example of Hooks Used in Planetary System Application 83
Table 6-16 Hook Template for Inheritance AnyActorHook 84
Table 6-17 Hook Template for Aggregation AnyMechanismHook 85
Table 8-1 Summary of Major Results 106
Table D-1 Overview of BOs and IOs in Knowledge Discovery Application 174
Table D-2 Overview of BOs and its IOs in Discovery of a Planet Application 183
Table D-3 Overview of BOs and IOs in Credit Card Fraud Detection Application 190
Table D-4 Overview of BOs and Corresponding IOs in Moviegoer's Application 193

CHAPTER 1 Introduction

We live in a digital world. Information technology has grown in a large scale in the past few years. Data is ubiquitous. Researchers have many instruments at their disposal that can be used to automatically, accurately, and quickly collect useful data. With the help of advanced technology such as bar code readers, scanners, radio frequency identifiers, and scantron forms data can be collected in less time compared to when data was only available manually. Now that data is collected using advanced technologies, another problem has emerged, namely how can useful information be extracted from data. In the past, when data was collected manually, the amount of data was manageable. A person could deal with the data intuitively, by simple examination, or by graphing. Now things have changed and the volume of data makes organizing and identifying useful information extremely difficult, if not impossible. As a result, researchers are encountering issues such as how can data be intelligently utilized, organized, and well distributed so that maximum benefit can be obtained. In other words, how can one look for the plant distribution rules in a tropical forest when one is a part of the forest itself? A person can touch every tree and flower in the forest, but still will not be able to visualize the entire distribution plan. However, if one views the forest in a helicopter from above, this tedious work becomes simple and straightforward. Data mining is the helicopter above the data forest. It allows one to view the data as a whole and extract usable information that could not have been obtained by simply browsing the data.

Data mining is one of the fastest growing fields in the computer industry. Once a small interest area within computer science, it has quickly expanded into a field of its own" [1]. Even though the concept of data mining has been a part of information technology for more then ten years, there are no established road maps or procedures that have been formally identified to guide the researcher. Researchers develop new algorithms and software manufacturers automate existing algorithms. As a result, the process of data mining has become increasingly complex. Data mining with the help of a Software Stability Model (SSM) [2,3] will serve as a powerful guide in the pursuit of data mining goals. The patterns named Discovery and Any Data Mining developed with the help of SSM will make the concept of data mining process simple and easy to understand. Other patterns such as Knowledge, Analysis, and Any Data Collection [4] are based on the same SSM paradigm. These patterns can be used to develop a tool that will flexibly adapt to any environment. These patterns are described in detail in the following Chapters.

1.1 Introduction to Data Mining

"Data mining, as a term, is the exploration and analysis, by automatic or semiautomatic means, of large quantities of data in order to discover meaningful patterns and rules" [5]. Data mining is most useful in exploratory analysis scenarios. "Data Mining is an iterative process within which progress is defined by discovery, through either automatic or manual methods" [1].

One of the greatest strengths of data mining is demonstrated in its wide variety of methodologies that can be applied to a host of problems in different domains. "Since data mining is a natural activity to be performed on a data warehouse, one of the largest target markets is the entire data-warehousing" [6,7], "data mart, and decision support community encompassing professionals from various industries, such as manufacturing, telecommunication, health-care, insurance, and transportation" [1]. Decision Support Systems, based on Database Systems, involve volumes of data, which is used for analyses and creating reports. "A data mart is a database, or collection of databases, designed to help managers make strategic decisions about their business" [8]. "A data warehouse is a collection of data designed to support management decision-making" [8]. "Data warehouses include a broad variety of data that present a consistent picture of business conditions at a single point in time." "Data warehousing combines databases across an entire enterprise; data marts are usually smaller and focus on a particular subject or department [9]."

1.2 Representation of the Problem

Researchers have developed many new and complex algorithms, which in turn, have put users in a dilemma as to which algorithm is best suited for their use. Algorithm selection is one of the most significant issues to be considered when data mining. Also, which algorithm is used in what application and in what situations is one of the important problems. No heuristics exist to determine the use of an algorithm in a particular domain or an application.

There is an uncertainty as how the efficiency can be improved or affected. This uncertainty arises because all the tools available on the market implement one or two of the available data mining algorithms, so efficiency of the output cannot be compared using the same tool.

Another significant issue that needs to be addressed is how to collect and prepare data so that it meets the algorithmic requirements. New collection technologies are constantly being introduced in the market. However, these collection methodologies [4] implement techniques that collect data from only a subset of the available sources. For example, with the introduction of Radio Frequency Identifiers (RFIDs) the collection system had to be changed in order to incorporate this new collection technique. It is challenging to change an entire system for the sole purpose of incorporating a new technique. If a new technique needs to be applied, it is important to investigate how productive the system can be.

In addition to the issues presented above, the discovery mechanisms for data mining are complex and have different categories such as scientific discovery and planet discovery. One of the discovery mechanisms used for data mining is knowledge discovery [10,11]. However, discovery mechanisms have more information to provide than just knowledge discovery of databases. These are some of the issues that became apparent when the topic was studied in detail. This research study is an attempt to create a pattern language that will address the issues presented above.

1.3 Software Stability Model Overview

The software stability modeling technique is best suited for building patterns because the patterns modeled are stable, easily adaptable, and can be exercised across many domains. Using the SSM course of action, one can identify and describe common segments in all applications, which will remain stable over time in cognitively similar applications. Patterns are divided into two categories for better perception: goals and capabilities [12]. These goals and capabilities shape the core knowledge of any application. These patterns go hand in hand and form mini architectures. The Stable Architectural Patterns that are built on these concepts can each be functional across many domains and can be employed in various applications. Various applications can be built on top of these architectures; these architectures will provide hooks [13] that will allow them to be incorporated into any application without much effort.

Figure 1-1 is a pictorial representation of the scope of applications built on top of the Knowledge Map (KM) also called Architectural Patterns. The core knowledge consists of the goals and capabilities. This core knowledge is adaptable and can be used across many domains without any modification. The Stable Analysis and Design Patterns together form the Architectural Pattern. This KM can be used in conjunction with other remote KMs to obtain full utilization of the core knowledge. More information about KM is provided in Chapter 5. Hooks are provided with the BOs to facilitate integration of the applications with the architecture. The scope of the application is enormous, and Figure 1-1 depicts how the scope can grow without changing the core knowledge solution to the problem.

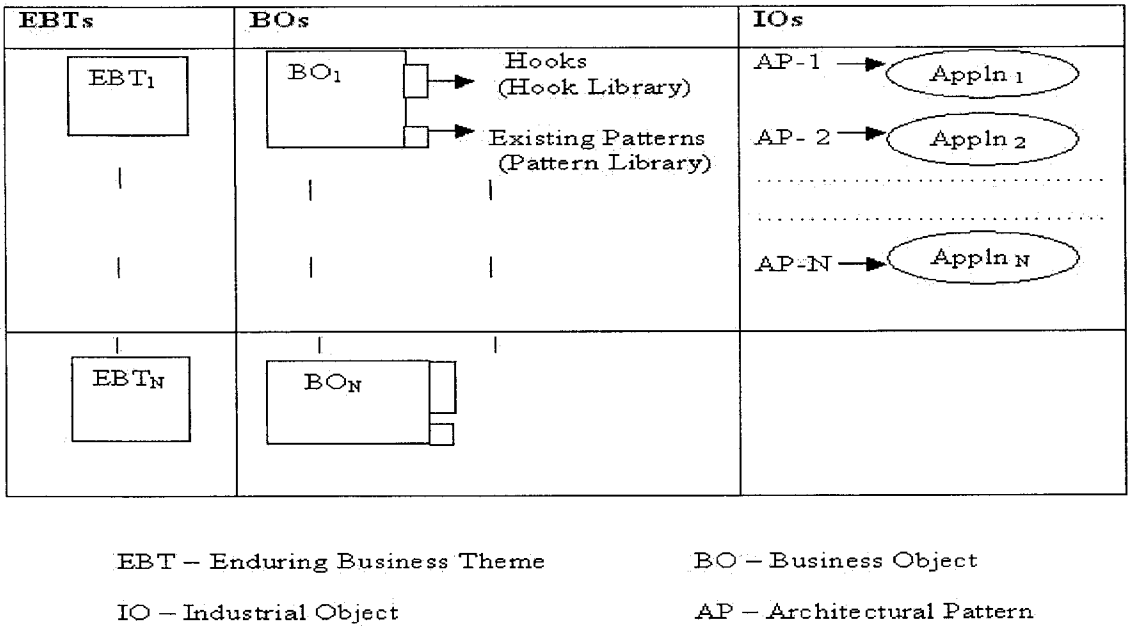


Figure 1-1 Scope of Applications Using the SSM

The solution to the problems stated above can be achieved with the help of the software stability paradigm and Pattern Language [2 – 4]. With this approach in mind, patterns have been generated that will solve the problems stated above.

Any Algorithm Selection pattern provides information that maps algorithms to domains and applications. The pattern provides features to select and add algorithms and mechanisms. This enables the pattern to be both stable and reusable over time.

Any Data Mining pattern provides the steps involved in data mining. It provides features that support the different varieties of data mining processes. It also provides the ability to add any new data mining mechanisms or algorithms.

Any Data Collection pattern considers the different data sources from where the data can be collected. It also ensures that any newly introduced data source or data collection technique does not change the pattern, but still provides the capability to add

these techniques. Any Data Preparation pattern supports all new and existing preparation techniques.

Discovery pattern provides the user with the ability to discover knowledge from a database or any other source. This pattern also supports the use of different algorithms using the Any Data Mining pattern for the same scenario, compare the results, and achieve maximum productivity and increase efficiency by using the best data mining algorithm suited for the application. The tool provides hooks to support integration of new techniques or new algorithms. Hooks act as classes that provide the framework with the capability to adapt to the required application.

This feature will give the user the ability to add new techniques without changing the core knowledge or the framework of the architecture. This modeling technique is explained in detail in the next section.

1.4 Research Methodology

The research methodology is based on the Stable Pattern Language (SPL). SPL is a language that consists of a group of patterns. These patterns are based on functionality and nature and are referred as Stable Analysis Patterns and Stable Design Patterns. Stable Analysis Patterns are also called goals of the system and the Stable Design Patterns are called capabilities of the system. These patterns are described in detail in the Chapters 3 and 4. SPL has many advantages over the traditional view (waterfall model) that will be discussed throughout the thesis. In contradiction to the long-established view of software development, where Testing only occurs after Analysis, Design and

Implementation, we believe that a major amount of time and effort will be saved if Testing is done earlier in the cycle. Stable Pattern Language integrates Testing with each of the software development steps of Analysis, Design, and Implementation. As a result, the developer is assured that the customer requirements are fulfilled and tested throughout the development cycle, and that no time or effort is spent Re-engineering or Testing. In addition, the old models such as the waterfall model require that the steps in the development cycle be performed in a sequential manner. This is in direct contrast to the SSM, which allows the model to be used in any application irrespective of the domain.

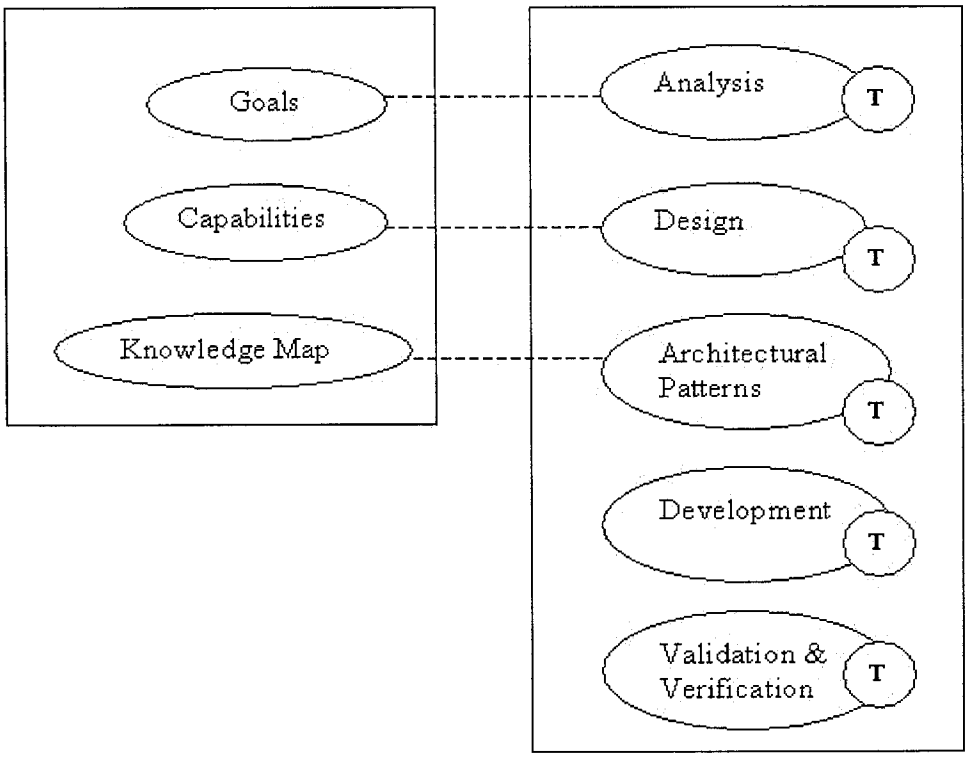


Figure 1-2 Stable Pattern Language - Testing

This research methodology also involves the following concepts:

Goals

Goals are the Stable Analysis Patterns [14], which are used us to analyze the problem and provide a solution, which will be stable over time. Additional information about goals is provided in Chapter 3.

Capabilities

Capabilities are the Stable Design Patterns [14]. These capabilities are the properties of the system, which are used to accomplish goals. More information about capabilities is provided in Chapter 4.

KM

For every goal there is a set of capabilities, which goes with it. KM gives a summary of how the goals and capabilities are interconnected. KM for data mining is discussed in Chapter 5.

Development

Depending on the KM generated, architecture is generated. The architecture provides hooks [13] that allow developers to use this architecture in any application and, in most scenarios, across many domains. Implementation details and development scenarios are discussed in Chapter 6.

Deployment (V and V)

During deployment developers validate and verify the architecture to check whether it meets all the requirements. They also ensure that the system meets the quality

factors during this step. Information about deployment, verification, and validation is detailed in Chapter 7.

1.5 Conducting the Research Methodology

Utilizing the approach of SSM to implement data mining concepts is very challenging. It involves the in-depth study of both the topic and the concepts. It is very confusing as to which concept will be stable and which will be unstable over time. Identifying the correct Enduring Business Themes (EBTs), Business Objects (BOs), and Industrial Objects (IOs) is another hurdle. During this research study, many scenarios were considered and the concepts of pattern language were analyzed to determine how they fit with data mining concepts.

Thesis Contributions

This thesis will serve to clarify data mining principles and will enable users to deal with data in a more knowledgeable manner. The thesis will illustrate the use of pattern language to deal with concepts of data mining. It will offer:

- Stable Analysis Patterns for Discovery, Knowledge, and Analysis
- Stable Design Patterns for Any Data Mining, Any Data Collection, Any Data Preparation, and Any Algorithm Selection
- A KM that explains how Analysis and Design patterns correlate with each other and form architectures
- Two applications for the Discovery pattern with hooks provided to adapt it to diverse applications

- Two applications for the Any Data Mining pattern with hooks provided to adapt it to different domains
- A demonstration to illustrate the applicability of these patterns.

1.6 Thesis Layout

The remainder of the thesis is organized as follows: Chapter 2 provides the comparative analysis of the data mining tools and pattern languages, Chapter 3 provides a description of the goals of the system, Chapter 4 describes the capabilities of the system, Chapter 5 introduces the KM, and Chapter 6 describes the implementation details and development scenarios. Chapter 7 describes the deployment scenarios, followed by the conclusions in Chapter 8. The report also contains appendices that describe the problem statement and present the code developed using the tool.

CHAPTER 2 Software Stability Model and Stable Pattern Language

Stable Pattern Language (SPL), the main building block of software stability, [14] has been a part of information technology for over a decade. Applying the concepts of software stability can reduce the cost of application development by reusing the architecture. This feature gives the developer time to develop more software, more time to market the product, and reduces the risk involved in development because the model has already been tested. Using the concept of software stability a developer can identify, adapt, and tailor business tasks and business capabilities. The scope of the applications is unlimited; as a result, the pay off is very high and generates a very high Return On Investment (ROI). These are some of the reasons that encourage developers to practice software stability.

Software stability consists of three layers: EBTs, BOs, and IOs [2 - 4]. EBTs are classes that provide the core knowledge about business and industry applications. BOs are classes that give access to this knowledge, and IOs are the objects that help the developer to realize knowledge in real world applications. These concepts will be explained in detail in the next section.

2.1 Pattern Language Evolution - Stability Concepts and Pattern Languages

Software stability and SPL are described using Table 2-1, which describes three research methodologies: Marketing, Stability, and Stable Patterns. These methodologies are defined according to different interest areas. For example, EBTs are known as goals

or quality factors in Marketing, but they are called Stable Analysis Patterns in SPL. Table 2-1 lists the different terminologies.

Table 2-1 Research Methodology Overview

Marketing	**Stability**	**Stable Patterns**
Goal or Quality Factor	EBT	Stable Analysis Pattern
Capability	BO	Stable Design Pattern
Mini Architecture	EBT and BO	Stable Architectural Pattern
Scenario Development	IO	Process Pattern
Quality Factor and Recommendation (V and V)	EBT + BO	Stable Analysis Pattern, Design Pattern, Architectural Pattern
Dynamic Analysis or The Business Language.	Stability Model or SW Development	Building Systems of Patterns

Encapsulating the core of the problem domain is mandatory prior to designing the right solution. To understand the problem it is first necessary to understand in detail the core knowledge and the concepts involved in software stability and SPL. Each of the term in Table 2-1 is briefly described below.

The innermost layer is the EBT layer; also called a Stable Analysis Pattern [2]. "Analysis is a tedious and time-consuming activity and efforts to accurately analyze any information makes the development of effective and reusable analysis artifacts of great interest" [15]. Stable Analysis Patterns form a promising base for facilitating and improving the quality of data analysis. The main objective of Stable Analysis Pattern is to define a model that captures the core characteristics of the problem that must be addressed.

The middle layer, the BO layer, is also called the Stable Design Pattern [2]. It is also called capability that describes the property of the system. BOs are the core artifacts or the workhorses of the system that change only externally and remain stable internally. BOs are adaptable through internal changes, and are semi-tangible that can be realized through experience or intuition.

The outermost layer is the IO layer. An IO lies towards the periphery of the system and is the most unstable. An IO is an application object; it is tangible and changes over time. It is the object that can be easily replaced or substituted as time changes. It is not adaptable and is explicit in nature.

The correlation among EBTs and BOs results in the architecture of the system, known as KM or Stable Architectural Pattern. A KM provides an overview of the logical flow between EBTs and BOs, and provides the developer with the tool to identify the various valid paths in the architecture. Hooks are attached to BOs in order to provide access points for building various applications using IOs.

An application must be verified and validated for its integrity and checked to ensure that it fulfills customer requirements. This is done with the help of the goals and capabilities. Testing can be integrated into each step by adding EBTs for Testing. Every goal and capacity has its own assessment mechanism. The Testing EBT is connected with other EBTs and is an integral part of the same architecture. EBTs are also connected to BOs. With the use of various architectures that are a part of the KM and the various patterns that are modeled, the system is then deployed for actual use. Quality factors are also taken into consideration. This is discussed in detail in Chapter 7.

2.2 Description of Existing Tools

This section describes the CART and XPertRule tools. CART supports Decision Tree Mechanism, Tree Navigation, and Navigation Reports, whereas XPertRule supports Tree Induction, Interactive Induction, and Pattern Exploration and Validation Tool.

2.2.1 CART

CART, a product from Salford System, is one of the state-of-the-art data mining tools available on the market. "CART is a robust, easy-to-use Decision Tree tool that automatically sifts large, complex databases searching for and isolating significant patterns and relationships" [16]. Decision Trees are tree-shaped structures that represent sets of decisions. These decisions generate rules for the classification of a dataset. Specific Decision Tree methods include Classification and Regression Trees (CART) [17 -19] and Chi Square Automatic Interaction Detection (CHAID). CART and CHAID are Decision Tree techniques used for classification of a dataset [20]. They provide a set of rules that one can apply to a new (unclassified) dataset to predict which records will have a given outcome. "CART segments a dataset by creating the 2-way splits while CHAID segments using chi square tests to create multi-way splits. CART typically requires less data preparation than CHAID. This discovered knowledge is then used to generate reliable, easy-to-grasp predictive models for applications such as profiling customers, targeting direct mailings, detecting telecommunications and credit card fraud, and managing credit risk [16]."

In addition, CART is an excellent pre-processing complement to other data analysis techniques. For example, CART's outputs (predicted values) can be used as inputs to improve the predictive accuracy of neural nets and logistic regression [16]. CART implements Decision Tree as discovery mechanism [21,22]. Figure 2-1 is the pictorial representation of the Decision Tree mechanism used in CART. This tool allows easy customization of trees and allows trees to be printed and exported for inclusion in reports and presentation. Figure 2-2 represents the tree navigator, which interactively explores a model in order to reveal key drivers and potential data errors. Figure 2-3 represents the tree summary reports provided by CART.

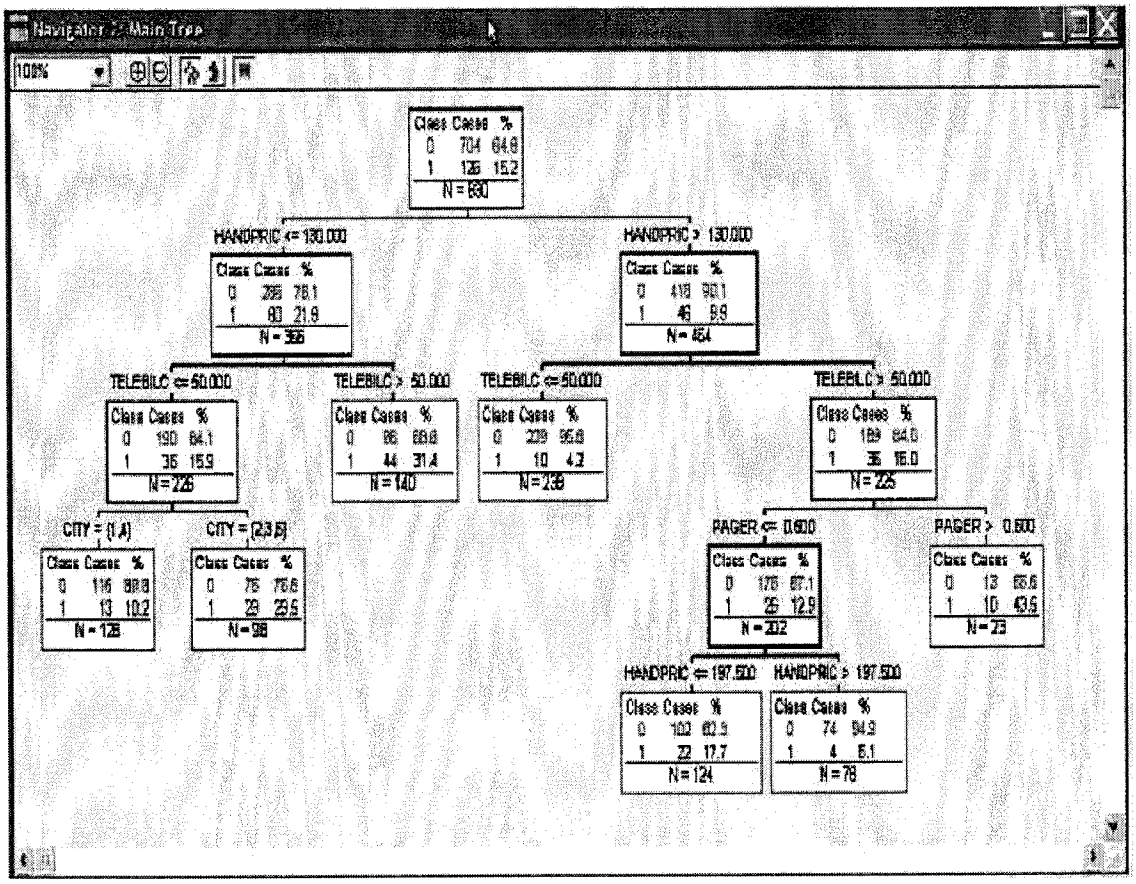


Figure 2-1 CART's Decision Tree Mechanism [23]

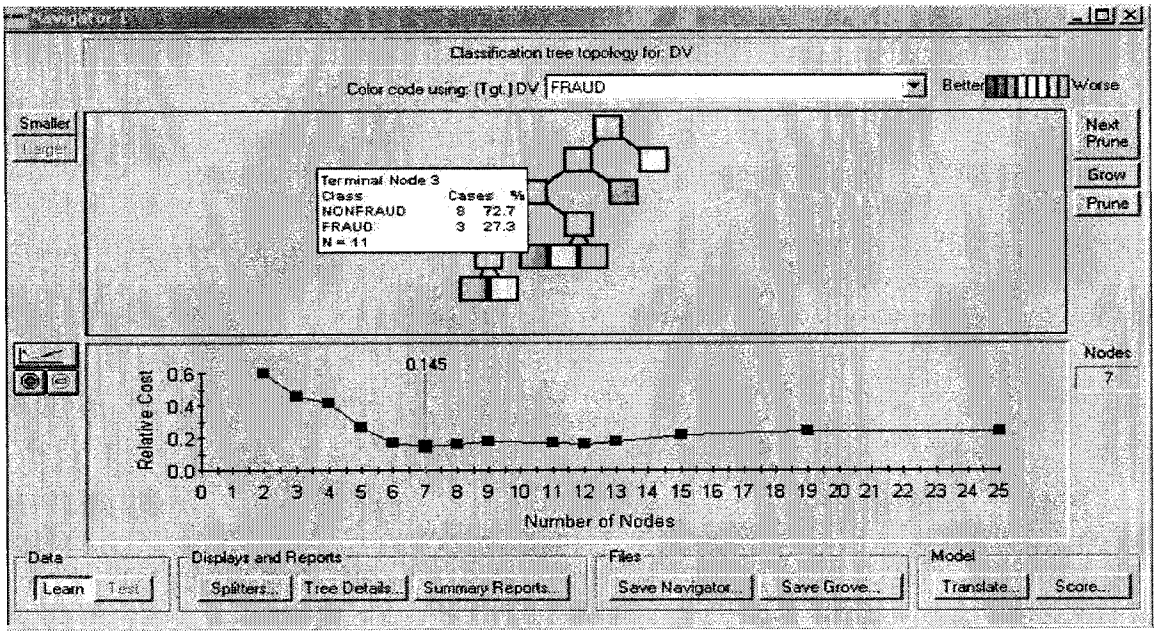


Figure 2-2 CART's Tree Navigator [23]

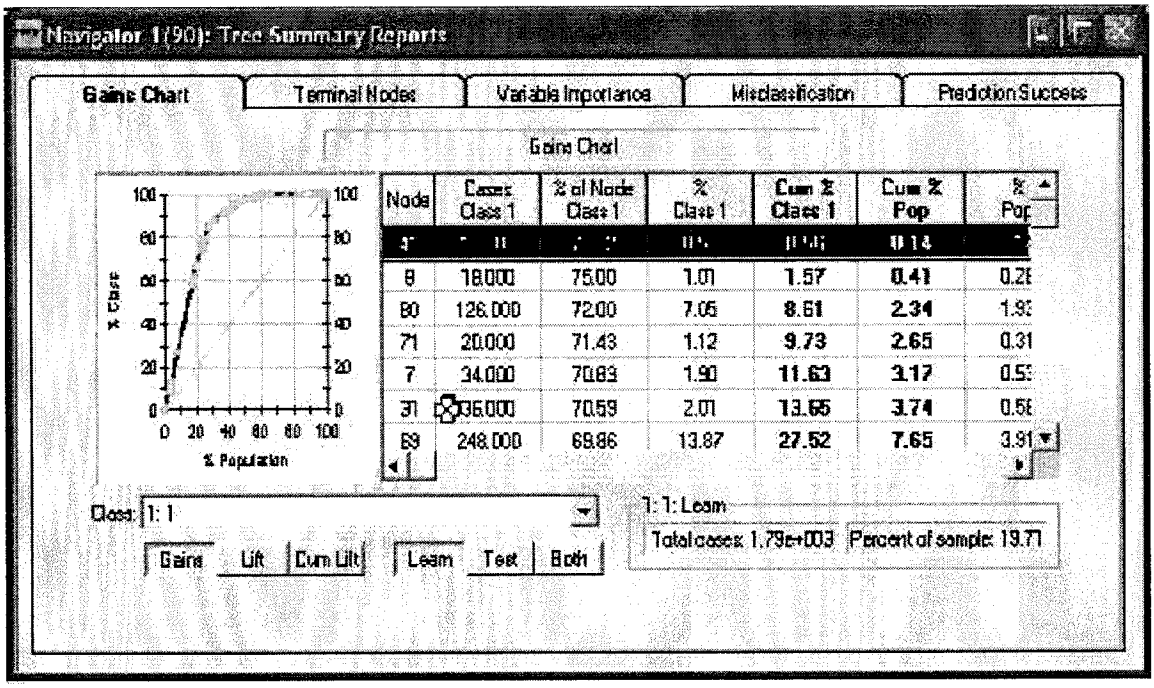


Figure 2-3 CART's Navigator Reports [23].

2.2.2 XPertRule

The information below is copied from the white paper "Data Mining with XpertRule." XpertRule Software is a member of the Special Interest Group set up in conjunction with the CRoss Industry Standard Process for data mining (CRISP-DM) [24]. The CRISP-DM project has developed an industry-neutral and tool-neutral data mining process model [25]. It is reassuring to see a common data mining methodology beginning to emerge. There is broad agreement on the main tasks within such a process: data preparation, data exploration, pattern discovery, pattern validation, and pattern deployment.

XpertRule Miner provides a graphical environment for supporting all the stages of the data mining process. In order to address industry wide data mining needs, XpertRule Miner supports many data mining techniques [26 - 29] such as induction. Tree Induction is a goal driven discovery and is the most widely used technique involving the induction of patterns (trees) relating to a business event (goal), such as mortgage arrears, customer attrition, energy consumption, and insurance claims. Figure 2.4 shows the pictorial representation of the Induction Tree mechanism [30,31] used in XpertRule Miner.

XpertRule also has interactive or incremental data mining: This combines automatic Tree Induction and manual tree construction. It enables the business user to develop tree patterns in collaboration with the induction algorithm. At every node (branch) in the tree, XpertRule Miner shows the importance of the various attributes at that point. The user is given the opportunity to impart their background business

knowledge and influence the choice of attribute splits while respecting the information evidence provided by Miner. Figure 2.5 illustrates interactive or incremental data mining.

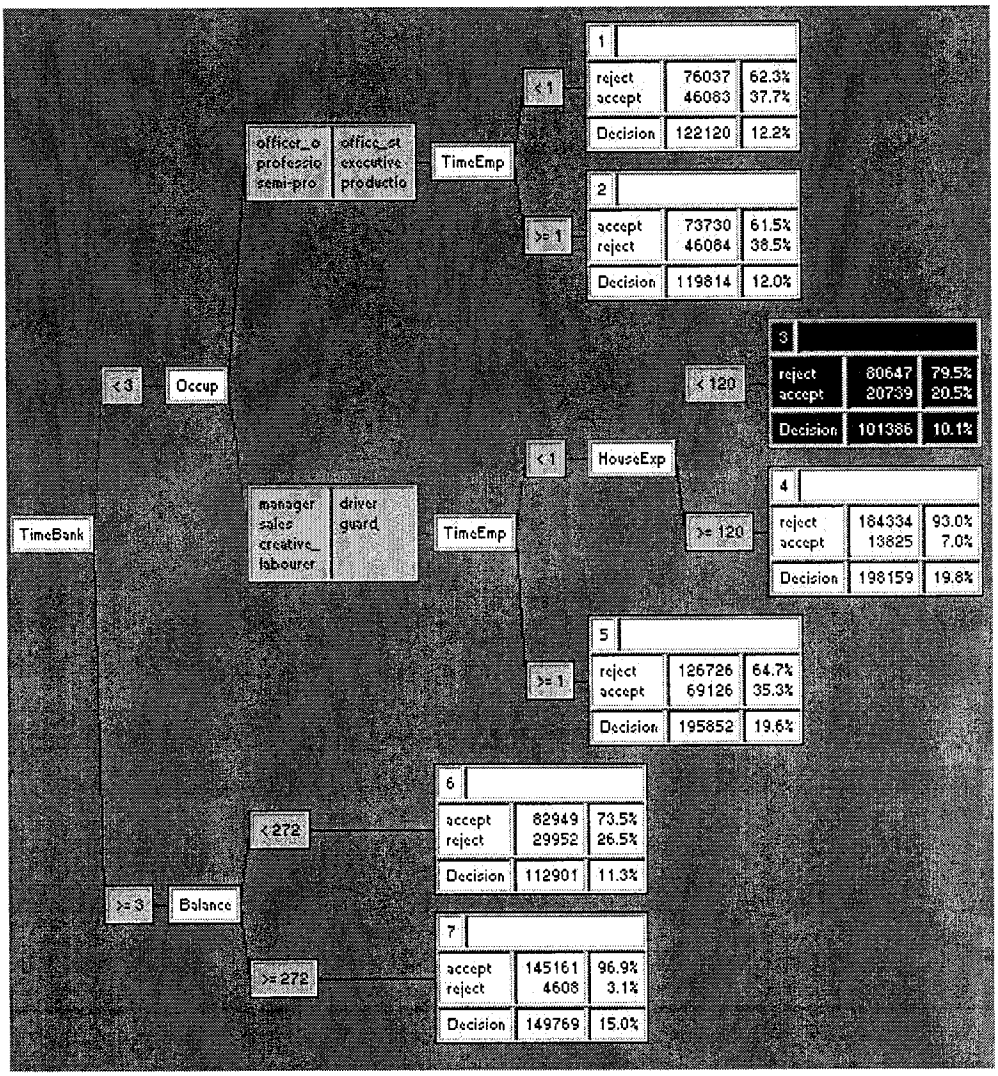


Figure 2-4 XPertRule's Tree Induction [24]

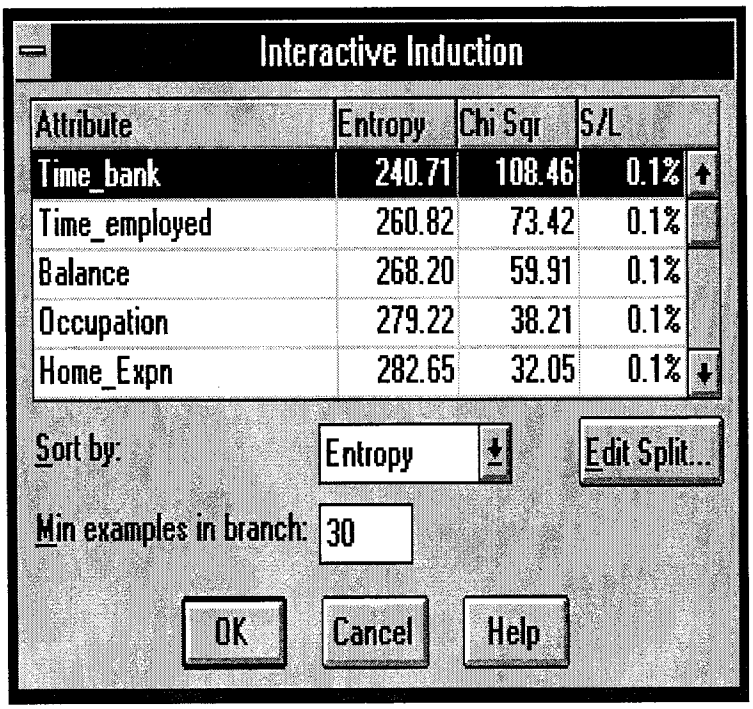


Figure 2-5 XPertRule's Interactive Induction [24]

XpertRule Software also provides a pattern exploration and validation tool. This is described in Figure 2-6. XpertRule Miner claims that Data Visualization and Exploration play an important role throughout the data mining process. During the Tree Induction process, XpertRule Miner allows user defined reports and data graphs to be updated dynamically as the user explores the various nodes and leaves (profiles) of the discovered tree. In addition to giving the user a method of validating the accuracy and meaning of tree patterns, the pattern exploration process helps the user obtain a better understanding of the patterns being discovered and their implications. XpertRule Miner supports a number of tree exploration reports: field statistics, frequency distribution, field propensity or value across profiles, and "gain or lift" graphs.

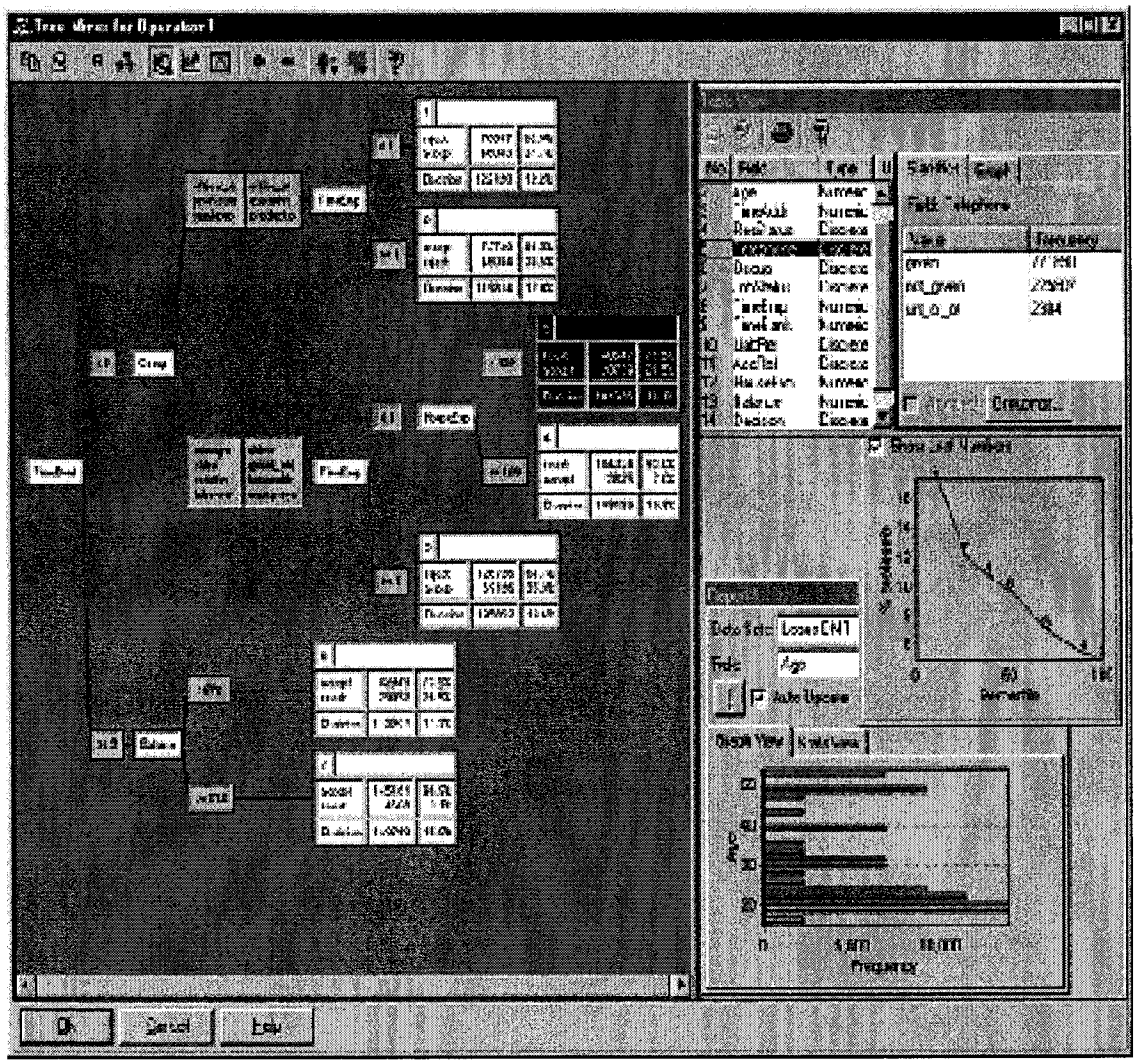


Figure 2-6 XPertRule's Pattern Exploration and Validation Tool [24]

2.3 Unified Data Mining Solutions

Unified Data Mining Solutions (UDMS) is the provisional name given to the tool presented in this thesis. The tool uses software stability concepts and the SPL paradigm. The pattern based tool implements patterns such as Any Data Mining, Discovery, Any

Data Collection, Any Data Preparation, Any Data Mining Mechanism, and Any Selection. Following is the list of features provided by the tool:

- **Stability:** UDMS is stable in nature because it uses the concepts of software stability and Stable Analysis Patterns such as Discovery, Analysis, and Knowledge, and Stable Design Patterns such as Any Data Mining, Any Data Collection, Any Data Preparation, Any Selection, and Any Data Mining Mechanism.
- **Flexibility:** UDMS provides the flexibility to add new algorithms and use any or all the available algorithms. The pattern Any Data Mining Mechanism is responsible for the support and implementation of algorithms. The user can select an algorithm and the selected algorithm is activated with the help of extension points called hooks.
- **Hooks:** UDMS provides hooks to add new data mining mechanisms and attach IOs to BOs. For example, data mining framework can be used for knowledge discovery [26, 27] as well as for data prediction, or data estimation.
- **Applicability:** UDMS is applicable in any data mining application irrespective of the domain. For example, the data mining framework can be used in credit card fraud detection applications, knowledge discovery [28, 29], genetic science, and medicine.
- **Web-based:** UDMS is web-based application. It uses Apache Tomcat Web Server, and java technologies such as Servlets, Java Server Pages, Core Java, and Java Database Connectivity.
- **Data Mining Processes Support:** UDMS provides support for all data mining processes. For example, support for data exploration, discovery of data patterns, collection, preparation, and selection is provided.

- **Data Mining Techniques Support:** UDMS provides support for all data mining techniques: data farming, data preparation, data collection, and data analysis.

2.4 Comparative Analysis of Tools

Three tools are compared and analyzed in this section: CART, XpertRule, and UDMS. The following criteria are used to compare and analyze the tools:

1) Reusability – demonstrates that the tool is reusable.

2) Applicability – checks the applicability feature of the tool. It lists the different applications in which the tool can be used.

3) Adaptive Features – compares the adaptive features of the tools. It checks whether the tool can adapt to different application with ease or whether it requires significant effort.

4) Core Knowledge – represents the core knowledge of the tool.

5) Stability – checks if the tool is stable to introduction of new algorithms or techniques.

6) Support for processes – represents the support the tool provides for the different available processes.

7) Support for algorithms – checks the number of algorithms the tool supports.

8) Level of Exploration – represents the level of exploration provided by the tool.

9) White Papers and Resources – checks whether any resources are available for reference.

10) Level of Data Mining Techniques – represents the level of support the tool provides for data mining techniques.

2.4.1 Reusability

This section compares the three tools using reusability as the criteria.

2.4.1.1 CART

CART implements a Decision Tree algorithm. It is a state-of-the-art classification tool that, as a standalone package, can investigate any classification task and provide a robust, accurate predictive model. CART methodology solves a number of performance, accuracy, and operational problems that still plague many current Decision Tree methods, but it cannot be used in applications where Estimation is required [16].

2.4.1.2 XpertRule

Research revealed that the tool developed by XpertRule software uses a Discovery pattern, which they developed [24]. It uses data mining mechanisms such as induction. This Discovery pattern narrows the concept of discovery and this software limits the use of data mining to only one mechanism. The Tree Induction mechanism implemented as a part of the software is used in applications such as mortgage areas, customer attrition, energy consumption, and insurance claims [32].

2.4.1.3 UDMS

The discovery process consists of more than one discovery mechanism. In order to demonstrate reusability, discovery and data mining patterns are modeled. These patterns are built taking into consideration diverse domains. In addition, implementation performed as a part of this thesis reveals that they are reusable across different domains.

2.4.2 Applicability

This section compares the three tools using applicability as the criteria.

2.4.2.1 CART

CART is used in data mining scenarios where only Decision Trees are required. These Decision Trees can be Classification or Regression Trees.

2.4.2.2 XPertRule

XPertRule uses Tree Induction techniques and is used in applications such as mortgage areas, customer attrition, energy consumption, and insurance claims [32].

2.4.2.3 UDMS

UDMS uses patterns such as Any Data Mining Mechanism, Any Data Mining, and Discovery, which allows the tool to be used in any application across different domains.

2.4.3 Adaptive Features

This section compares the three tools using adaptive feature as the criteria.

2.4.3.1 CART

CART does not have the capability to adapt itself to any algorithm. It implements only one data mining mechanism, which is Decision Tree. With the introduction of a new data mining mechanism or, more specifically, a new Decision Tree mechanism, the tool is not adaptive.

2.4.3.2 XpertRule

XPertRule does not provide the implementation of any other data mining mechanism such as Neural Networks or Genetic Algorithms. When a new algorithm is introduced, the existing software must be changed to provide the new functionality [33].

2.4.3.3 UDMS

The features provided in Any Data Mining and Discovery patterns provide flexibility to add new data mining and new discovery mechanisms, thus making the tool adaptive in nature.

2.4.4 Core Knowledge

Core knowledge describes the internal stable knowledge. This section compares the three tools using core knowledge as the criteria.

2.4.4.1 CART

CART does not have any core knowledge: it simply implements Decision Tree mechanisms.

2.4.4.2 XPertRule

XPertRule uses Data Visualization as a part of the data mining component [33]. Data Visualization is not a part of core knowledge of data mining; rather Data Visualization is used to represent data mining in a pictorial fashion.

2.4.4.3 UDMS

UDMS consists of goals (EBTs) and capabilities (BOs) as part of core knowledge of data mining. This core knowledge contains patterns, but does not contain any specific algorithm or mechanism, which changes over time.

2.4.5 Stability

This section compares the three tools using stability as the criteria.

2.4.5.1 CART

CART is not stable in nature because it does not provide the feature to add new data mining mechanisms or algorithms.

2.4.5.2 XPertRule

XPertRule is not stable in nature because it is not pattern based and is not built to implement new mechanisms and techniques.

2.4.5.3 UDMS

UDMS is pattern based and uses software stability to build the patterns. As a result, the tool is stable in nature.

2.4.6 Support for Processes

This section compares the three tools with respect to support for processes.

2.4.6.1 CART

CART provides basic data exploration, extracting relationships, data associations, and model building.

2.4.6.2 XPertRule

XPertRule provides data exploration, data collection, pattern discovery, data selection, goal classification, pattern deployment, and pattern validity [11].

2.4.6.3 UDMS

UDMS provides data exploration, data preparation, data collection, pattern discovery, algorithm selection, goal classification, pattern deployment, pattern validity, and pattern verification.

2.4.7 Support for Algorithms

This section compares the three tools with respect to support for algorithms.

2.4.7.1 CART

CART provides support for Decision Tree algorithms such as Classification and Regression Trees.

2.4.7.2 XPertRule

XPertRule provides support for Induction Trees [33].

2.4.7.3 UDMS

UDMS provides support for all data mining algorithms such as Prediction, Decision Trees, Induction Trees, Genetic Algorithms, and other statistical methods.

2.4.8 Level of Exploration

This section compares the three tools with respect to support for level of exploration.

2.4.8.1 CART

CART does not provide basic data exploration through visualization, but provides discovery of data patterns.

2.4.8.2 XPertRule

XPertRule provides basic data exploration through visualization and discovery of data patterns [32].

2.4.8.3 UDMS

UDMS provides basic data exploration through visualization, provides discovery of data patterns, and provides data mining patterns.

2.4.9 White Papers and Resources

This section compares the three tools with respect to the resources available.

2.4.9.1 CART

CART provides the following white papers as resources to get more information.

- Salford Systems Data Mining 2006, http://www.salforddatamining.com/tutorials06.htm
- CART is a new tree structured statistical analysis and data mining tool, http://www.sciencedownload.com/Data_Analysis/Data_Mining/CART/
- Salford Systems, http://www.salford-systems.com/whitepapers.php
- Salford Systems, http://www.salford-systems.com/4215.php

2.4.9.2 XPertRule

XPertRule provides the following white papers as resources to get more information [11, 32, 33].

- White papers on Knowledge Management and Data Mining, http://www.xpertrule.com/tutor/papers.htm
- XpertRule Software Ltd, http://www.xpertrule.com/tutor/mining.htm
- White papers on Knowledge Management and Data Mining http://www.intellicrafters.com/mineroverview.pdf.

2.4.9.3 UDMS

UDMS will soon provide white papers as resources to get more information.

2.4.10 Level of Support for Data Mining Techniques

This section compares the three tools with respect to the level of support they provide for data mining techniques.

2.4.10.1 CART

CART provides functionality to clean up data, execute data transformations, handle null or missing data values, and perform model evaluation. CART doesn't provide Supervised Induction and Tree-based Clustering, but association discovery is under development.

2.4.10.2 XPertRule

XPertRule provides Supervised Induction, and Tree-based Clustering, but association discovery is under development. XPertRule provides functionality to clean

up data, execute data transformations, handle null or missing data values, and perform model evaluation [33].

2.4.10.3 UDMS

UDMS provides Supervised Induction, Tree-based Clustering and association discovery, and functionality to clean up data. UDMS executes data transformations, handles null or missing data values, and performs model evaluation.

2.4.11 Overview of Comparative Analysis

Table 2-2 provides an overview of the comparative analysis using the different criteria described in previous section.

Table 2-2 Overview of Comparative Analysis of Data Mining Tools

Tool Name / Features	CART	XPertRule	UDMS
Reusable	Not much	Not much	Highly
Adaptive	Not at all	Not at all	Highly
Stable	Not at all	Not at all	Highly
Core Knowledge	No	Tree Induction	Goals and capabilities.
Applicability	Restricted	Restricted	Across different applications and different domains.
White paper and resource availability	Yes	Yes	Yes

Table 2-3 provides the comparison of the support of tools for data mining algorithms. The 'X' represents that the tool supports the algorithms.

Table 2-3 Comparison of Tools for Algorithms

Tools / DM Algorithms	CART	XPertRule	UDMS
Clustering			X
Neural Networks			X
Genetic Networks			X
Classification Tree	X		X
Regression Tree	X		X
Chi squared			X
Text Mining			X
Web Mining			X
Statistical Algorithms			X
Induction Trees		X	X

Table 2-4 provides the comparison of the tools in terms of support for processes. The 'X' represents that the tool supports the processes.

Table 2-4 Comparison of Tools for Processes

Tools / Processes	CART	XPertRule	UDMS
Basic Data Exploration	X	X	X
Extracting Relationships	X		X
Data Associations	X		X
Model Building	X		X
Data Collection		X	X
Pattern Discovery		X	X
Data Selection		X	X
Goal Classification		X	X
Pattern Deployment		X	X
Pattern Validity		X	X
Algorithm Selection			X
Pattern Verification			X

Table 2-5 provides the comparison between the tools in terms of support for exploration processes. The 'X' represents that the tool supports level of exploration.

Table 2-5 Comparison of Tools for Exploration Process

Tools / Exploration	CART	XPertRule	UDMS
Basic Data Exploration through Visualization	X	X	X
Discovery of Data Patterns	X	X	X
Data Mining Patterns			X
Any Data collection			X
Any Data preparation			X
Any Algorithm Selection.			X

Table 2-6 provides the comparison between tools in terms of data mining techniques The 'X' represents that the tool supports data mining techniques.

Table 2-6 Comparison of Tools for Data Mining Techniques

Tools / DM Techniques	CART	XPertRule	UDMS
Clean Data	X	X	X
Data Transformations	X	X	X
Null or Missing Values	X	X	X
Model Evaluation	X	X	X
Supervised Induction		X	X
Clustering		X	X
Association Discovery		X	X

CHAPTER 3 Data Mining Goals

Building a system requires planning, defining the goals the system is supposed to meet, analyzing the goals, designing the system, and implementing it. SSM [2,3] introduces goals, a requirement, which every system should meet. Goals are defined as the purpose toward which an endeavor is directed. These are the classes that form the core concept. With the help of these goals developers build a system based on well-defined objectives. These goals are defined with the help of EBTs [3]. These EBTs provide the system a strong backbone on which the entire body is built. These EBTs are internally and externally stable, and they remain constant over time. They adapt themselves to any environment without any modifications, but they are intangible and cannot be easily realized. Their identification requires in-depth knowledge of the environment where the system is going to be deployed. These EBTs form a team with other EBTs, related to Quality and Testing, which measure the desirable quality factors and tests the system concurrently with the task of building it.

Underneath these goals are the subgoals. These subgoals are the subordinates of the goals or the EBTs, and are an integral part of the goals, and give us detailed information about the goals. They also depict the function of the goals. For example, if a goal is Documentation then the subgoals are Representation and Illustration. Representation and Illustration are a part of Documentation, but they reveal the vital functions that Documentation provides. Another example of the subgoal is

Brainstorming. These subgoals specify the role that Documentation plays, and the function it performs in the evolution of the system.

3.1 Overview of Goals

This section introduces the data mining goals, and provides the description and the UML models of each of the goals.

3.1.1 Discovery

The goal of discovery is to discover hidden patterns, trends, associations, anomalies, and statistically significant structures and events in data [34]. Discovery has a great impact on the strategies that can be employed to get better insight into the market and increase productivity. Figure 3-1 illustrates the UML model of the Discovery Stable Analysis Pattern. The pattern description is provided in the text following the diagram. The detailed documentation for the Discovery Stable Analysis Pattern is provided in Appendix A – Stable Analysis Pattern.

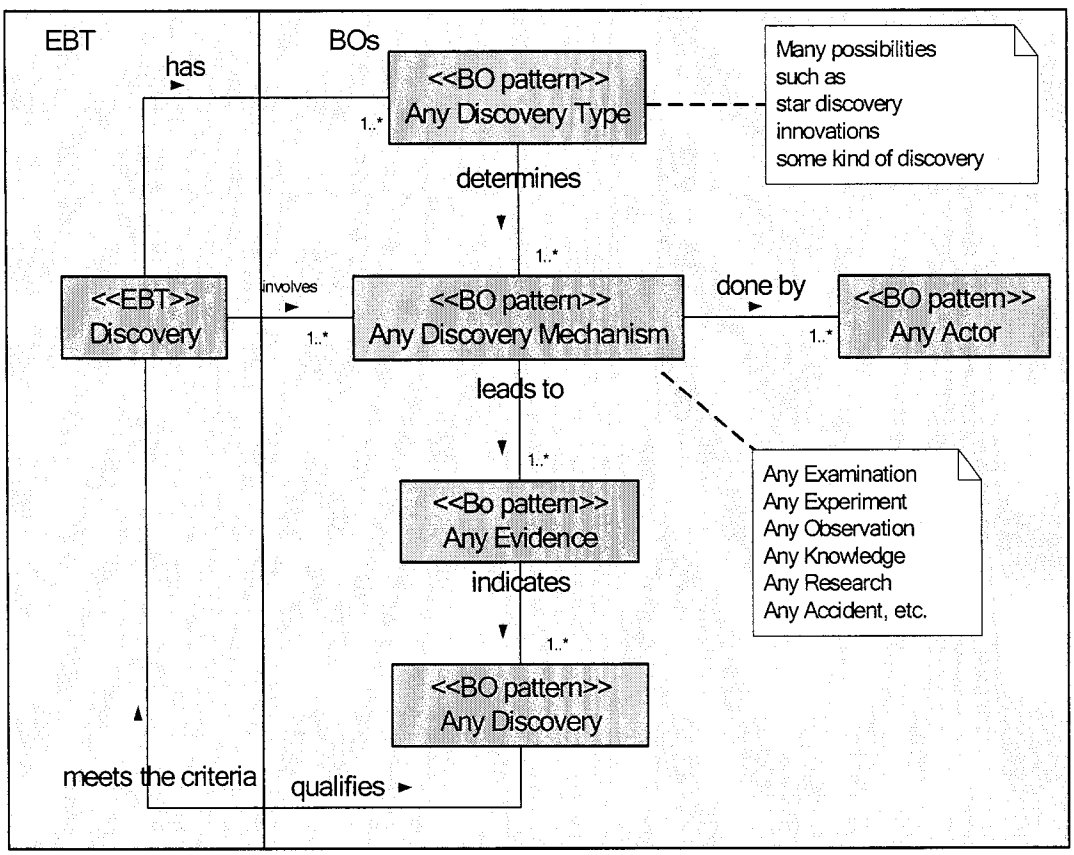


Figure 3-1 Discovery Pattern (taken from Dr. Fayad's patterns archive)

3.1.1.1 Discovery Pattern Documentation

The definition of discovery in Section 3.1.1 explains the discovery process. There are many different types of discovery and each has its own mechanism in its domain. For example, the discovery process in the medical area may involve Experiments, Observation, or Research. To realize all different discovery types Any Discovery Type pattern in introduced. Discovery can be discovery of anything. To give the word "anything" a proper semantic, the pattern name "Any Discovery" is chosen because the process of discovery leads to the final discovery. It's a different issue whether the result is acceptable or not.

To prove itself, every discovery requires evidence. The evidence can be proved through any mechanism. The mechanisms could be some Experiments, or some Observations, or some algorithms. As a result, we have Any Evidence and Any Discovery Mechanism in our Analysis pattern. Any Discovery Type determines the Any Discovery Mechanism. Depending on the type of discovery, its mechanism is determined. For example, for knowledge discovery the probable mechanisms are Clustering, Text Mining, and Prediction. The different mechanisms for discovering a planet could be Research and Observation. Any person or any organization is responsible for the discovery because they are the ones who observe or examine the result. Consequently, we have Any Actor in our model.

3.1.2 Knowledge

Knowledge is the awareness and understanding of facts, truths, or information gained in the form of experience or learning [24]. Knowledge is also defined as "information combined with experience, context, interpretation, and reflection. It is a high-value form of information that is ready to be applied to decisions and actions [24]." Figure 3-2 provides the UML model of Knowledge Stable Analysis Pattern. The pattern description and the pattern documentation are provided in the text following the diagram.

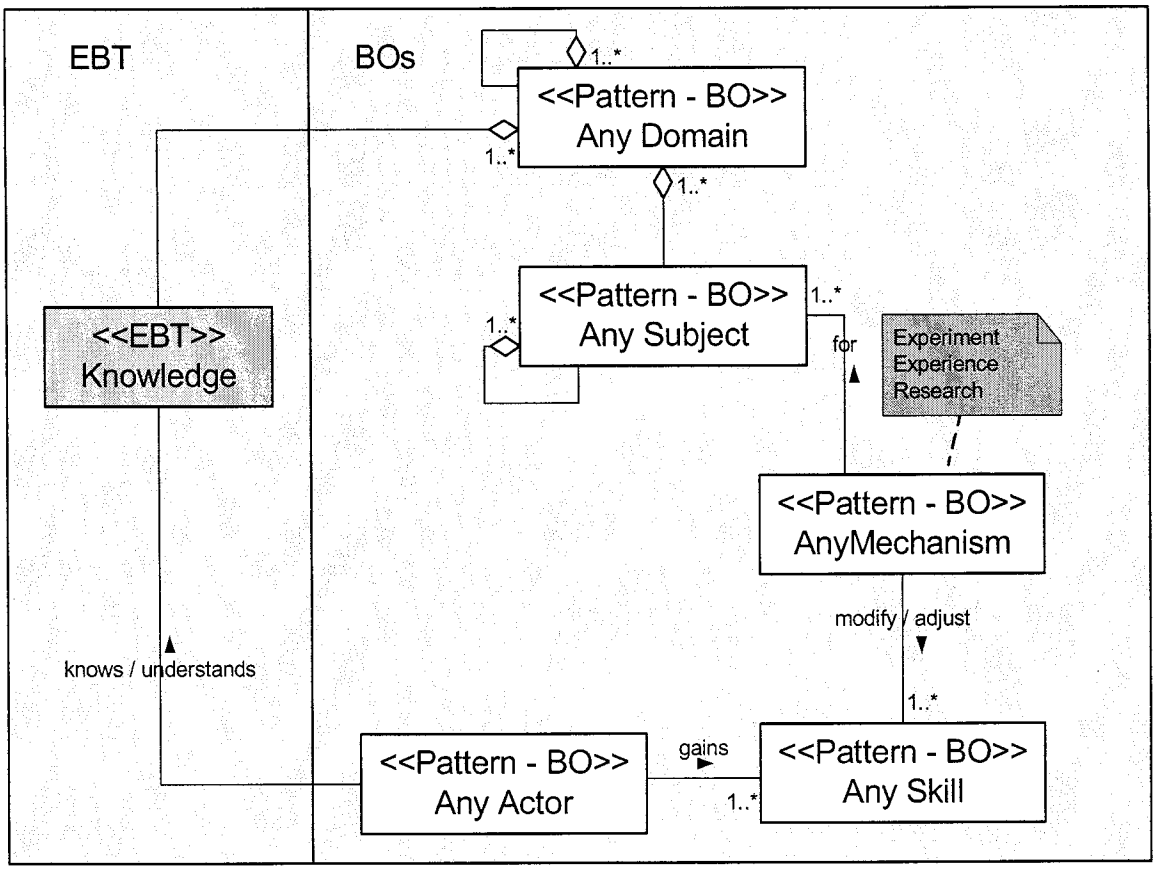


Figure 3-2 Knowledge Pattern (taken from Dr. Fayad's patterns archive)

3.1.2.1 Pattern Documentation

Pattern Name

Knowledge Stable Analysis Pattern

Context

The Knowledge Stable Analysis Pattern can be utilized in any scenario where knowledge is acquired or gained. This pattern can also be used in any domain or in any subject.

Problem

Knowledge refers to what one knows and understands. Knowledge is defined as the remembering of previously learned material. The problem faced here is that the different ways to achieve knowledge are not represented anywhere.

Knowledge can be structured or unstructured and implicit or explicit. Knowledge that is unstructured and understood, but not clearly expressed, is implicit knowledge. If the knowledge is organized and easy to share then it is called structured knowledge. Another problem is that knowledge is not modeled in a pattern because different domains use different kinds of knowledge.

Knowledge is a very broad term that can be used in different domains and in different subject areas. The problem encountered here is that no pattern or tool exists, that represents knowledge from the different domains and different subject areas in a concise manner.

Solution

1. Pattern Description

To represent different types of knowledge, we have Knowledge as a Stable Analysis Pattern. Knowledge can be used in one or more domains and subject areas; therefore, we have BOs Any Domain and Any Subject. To capture the different ways in which knowledge can be achieved, BO Any Mechanism is modeled. Any Actor is a BO, which represents the system, organization or any person acquiring knowledge. Any Actor also gains different skills to increase knowledge, or it uses different mechanisms to modify, or adjust the different skill, so we have BOs Any Skill and Any Mechanism.

2. Participants

Patterns

Any Domain – Represents the different domains where knowledge is present.

Any Subject – Represents the different subject areas where knowledge is present.

Any Mechanism – Represents the different mechanisms, which are used to acquire knowledge. Examples are Research, Experiments, Experience, and Memory.

Any Actor - Represents an organization, or a system, or any person acquiring knowledge.

Any Skill – Represents the skill acquired by Any Actor.

Class

Knowledge – Represents the specific information about something.

3.1.3 Analysis

Analysis is the process of systematically applying statistical and logical techniques to describe, summarize, and compare data [35]. It also means the systematic study of data so that the meaning, structure, relationships, and origins are understood. Examples of Analysis techniques are Exploratory Data Analysis, Market Based Analysis, and Memory Based Analysis Technique [4,19]. Figure 3-3 provides the UML model of Analysis Stable Analysis Pattern. The pattern description and the pattern documentation are provided in the text following the model.

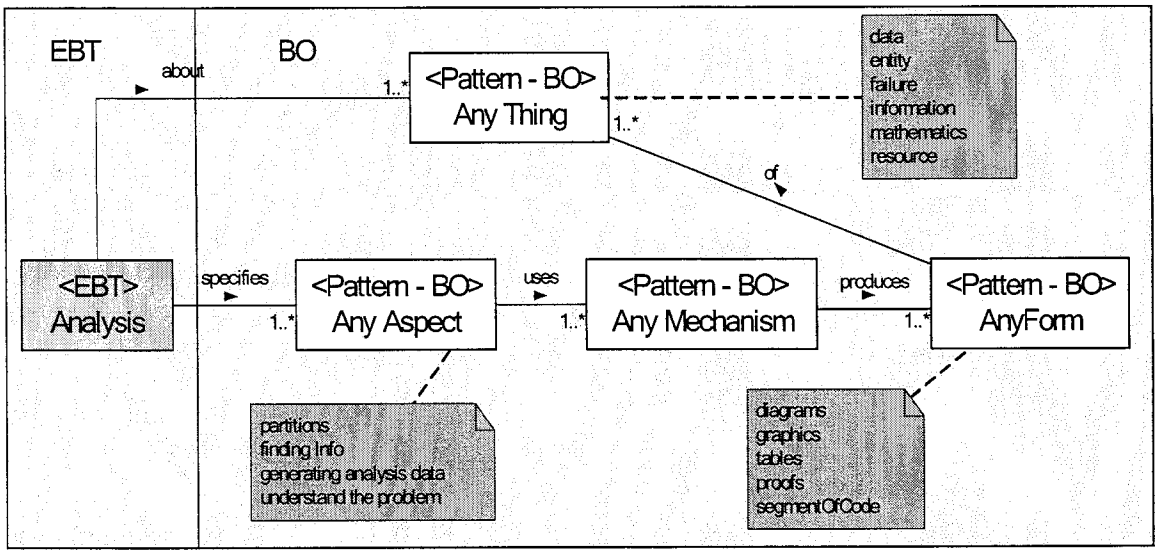


Figure 3-3 Analysis Pattern (taken from Dr. Fayad's patterns archive)

3.1.3.1 Pattern Documentation

Pattern Name

Analysis Stable Analysis Pattern

Context

This pattern can be utilized in any scenario where analysis of anything is required. This pattern can also be used in any domain irrespective of what is to be analyzed.

Problem

Analysis is the process of understanding an entity or idea by examining it in terms of its various parts. Usually analysis is considered in terms of analyzing data, but Analysis is also a branch of mathematics, which deals with real numbers, complex numbers, and their functions. Analysis in terms of physiology is an account of the meaning, or content of a word, phrase, or concept. Analysis is also defined as the process of identifying a question, or issue to be addressed, modeling the issue, investigating

model results, interpreting the results, or possibly making a recommendation. Analysis is a broad term, and in different domains has different applications. The main problem faced here is that an analysis pattern that can be used in any domain does not exist.

Solution

1. Pattern Description

Analysis analyzes any data, any entity, any object, and any information. To give all these things that Analysis can analyze a proper name, we have the BO Any Thing. Analysis also specifies the reason or aspect for analysis. For example, the aspect could be finding information, or generating analyzed data, or understanding a problem domain. As a result, we have BO Any Aspect. Since different mechanisms are required to find information or generate analyzed data, we have BO Any Mechanism. In addition, these mechanisms produce results that take any form. Therefore, we have BO Any Form.

2. Participants

Patterns

Any Thing – Represents the object, entity, or information, which is to be analyzed.

Any Aspect – Represents the different ways something can be viewed.

Any Mechanism – Represents the different mechanisms, which are used to analyze. Examples are Exploratory Data Analysis, Memory Based Analysis, and Market Based Analysis.

Any Form - Represents the form in which the results or Any Thing appear.

Class

Analysis - Represents the process of identifying a question or issue to be addressed.

3.2 Subgoals

Each goal described above has its subgoals. These subgoals give us a better understanding of the goal. However, these subgoals can also be responsible for distraction from the main goals. They can contaminate the goals by diverting attention to themselves and interfering with the main aspiration.

Let's examine the subgoals and examples of how they contaminate their respective goals.

3.2.1 Subgoals of Discovery

Discovery is a significant goal in data mining. The corresponding subgoals are ***Identification*** and ***Designation***. Identification means capability to find, retrieve, report, change, or delete specific data without ambiguity [36]. Sometimes Identification can hamper the purpose of discovery by looking for data, or information, which is already present and not some thing new. Also, **Exploration** is a subgoal. Exploration is defined as the search, using varied techniques, with the objective to discover and evaluate data [37]. ***Formularization*** is another subgoal of discovery. Formularization means the framework should provide a step-by-step guide and well-defined selections to the users [36]. The guidelines should be standardized and easy to understand. This

formularization will assist users to follow a series of steps to select the algorithm depending on the data and the end result to be achieved. Discovery is knowledge discovery process and not a process, which guides the user with steps to be performed. This shows how formularization can deviate our attention from our main objective.

Classification [17] is another subgoal of Discovery. Classification of data means the data that is collected needs to be categorized. Classification is done to predict the behavior of the data. Therefore, the subgoal Classification can have forms such as ***Categorization***, ***Distribution,*** and ***Prediction*** [38,39], but all of these forms are techniques used in discovering knowledge and have nothing to do with discovery. If we concentrate on these forms we are limiting ourselves to one area leaving the major objective.

3.2.2 Subgoals of Knowledge

Knowledge is one of the goals of data mining. Knowledge has subgoals such as ***Learning*** and ***Brainstorming***. Learning is defined as the process of acquiring knowledge, attitudes, or skills from study, instruction, or experience [40]. Brainstorming is defined as a problem-solving technique that involves creating a list that includes a wide variety of related ideas. Brainstorming and Learning are very closely associated with knowledge, but are not goals of data mining. Hence, they are categorized as subgoals.

3.2.3 Subgoals of Analysis

Analysis is one of the goals of data mining. Analysis has subgoals such as ***Investigation, Separation, Integration*** [41], and ***Differentiation*** [41]. Investigation is defined as a detailed inquiry or systematic examination of something. Also, Separation means separating or classifying elements. Similarly, Integration [41] and Differentiation [41] are the analytical techniques used in mathematics. All these subgoals are derived from the goal Analysis, but have different meanings.

CHAPTER 4 Data Mining Capabilities

Capabilities are the properties of the system that assist in fulfilling the goals. Capabilities are defined with the help of BOs [42]. BOs have some characteristics such as they are externally stable over time, internally unstable, adaptable through internal changes, and tangible. These characteristics help us to identify the capabilities of a system.

To understand and identify the capabilities of data mining, let's consider an example where a data mining application can be used to analyze data that is collected from various sources, and transformed to an understandable format. For example, let's consider a scientist, who is the head of the Research Department (HOD). This scientist has collected and analyzed data from her research project on patients. There are different kinds of data for each person, such as blood samples, urine samples, breast tissue samples, records from medical tests, and data generated from analyzing them. The HOD believes that there should be some correlation between the data from the patient and the chance of him or her contracting cancer. Although the HOD personally has a deep understanding of the data, the rules are not obvious and the amount of data is too large to analyze manually. Therefore, the HOD hopes that a computer can be helpful in finding some of these internal rules in her data and in testing some of the presumptions. Given this situation, the artifacts from this thesis on data mining can be of great assistance. Since the HOD understands little about these complicated algorithms, the data mining

application will select the algorithm that best matches the goals. So let's start analyzing the data mining process one concept at a time.

4.1 Overview of Data Mining Capabilities

This section provides the data mining capabilities and its description. This section also provides the pattern for some of the capabilities.

4.1.1 Any Data Collection

Information in the form of data is present everywhere in various formats, some in highly structured formats such as rows and tables, some semi-structured formats such as XML data, and some non-structured data. This data is gathered and combined from various sources [4]. Any Data Collection represents this process of collecting elements. Figure 4-1 illustrates the UML model for the Any Data Collection pattern. Pattern description is provided in the text following the diagram.

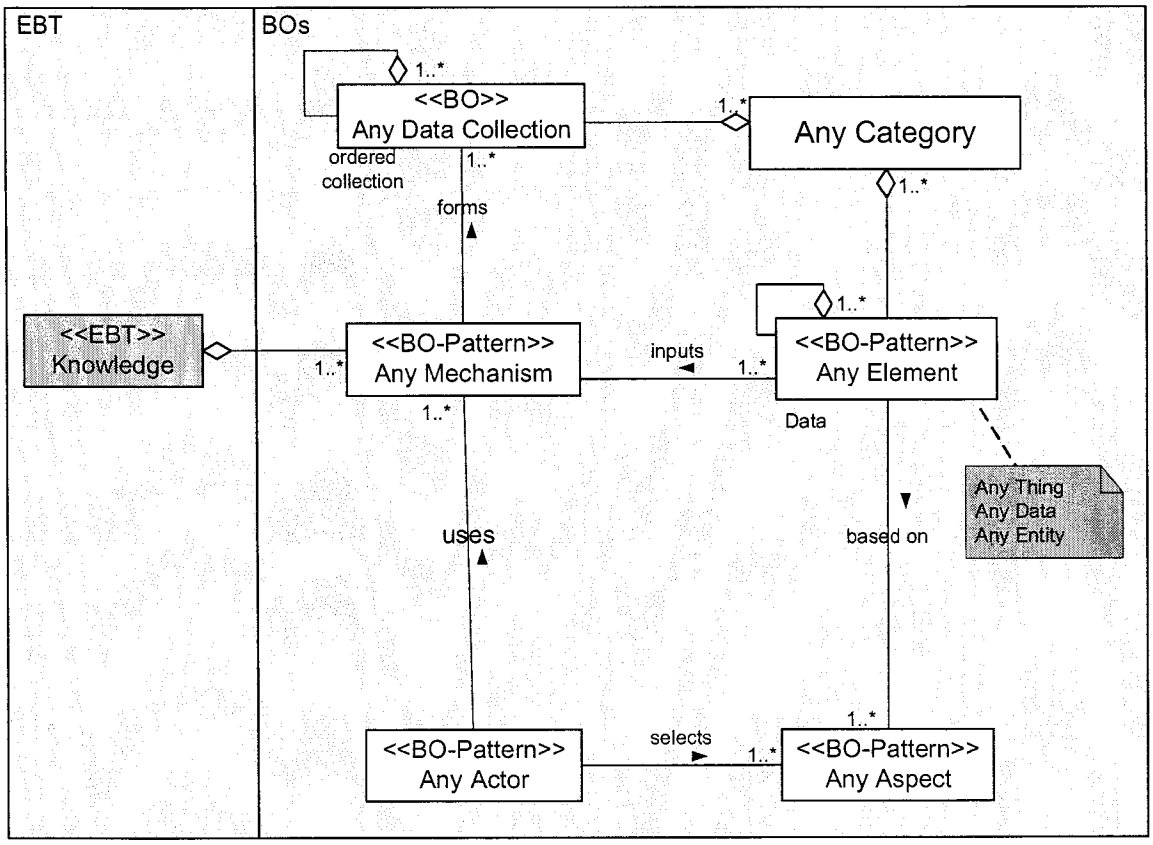


Figure 4-1 Any Data Collection Pattern (taken from Dr. Fayad's patterns archive)

4.1.1.1 Pattern Documentation

Pattern Name

Any Data Collection, as a term, is accumulation of elements; these elements can be any object or any data. Since collection is a broad term, the pattern is named Any Data Collection.

Context

This pattern can be used in any application where collection is needed. This is a very generic pattern, which is modeled taking into consideration various domains. As a result, this pattern can be used in any application.

Problem

Collection is the process of accumulation of elements. Some collections contain data, whereas some contain entities. No application or pattern exists, which represents a collection containing anything. Different collection mechanisms exist such as RFIDs, scantron forms, surveys, and questionnaires. Also, the available data collection tools implement only one or two algorithms to collect elements from sources. With the change of an existing algorithm or introduction of a new algorithm or mechanism, these tools need to be changed. This is a major problem faced by the current data collection tools.

Solution

1. Pattern Description

Any Data Collection or Any Collection is a collection of elements. These elements can be Any Data, or Any Entity, or Any Elements. Any Collection can be formed of other collections. For example, a library is a collection of books, which can also contain a collection of technology books, or mathematics books. Any Data Collection and Any Element are a part of Any Category.

Any Actor uses one to many Any Mechanisms to form Any Collection. Similarly, Any Actor selects Any Aspect based on which Any Element is selected. Any Element is also used as input to Any Mechanism.

2. Participants

Patterns

Knowledge – Represents the specific information about something.

Any Mechanism – Represents the techniques and algorithms used to collect Any Element.

Any Element – Represents the object that is collected. It could be Any Data, Any Entity, or Any Thing.

Any Actor – Represents Any System or Any Person, who plays an important role in Collection.

Any Aspect – Represents the different ways something can be viewed.

Any Category – Represents the different categories, which contains Any Collection and Any Element.

Class

Any Data Collection – Represents the collection process.

4.1.2 Any Data Mining

Data mining is an information extraction activity, whose goal is to analyze data and discover hidden facts contained in databases [5]. Using a combination of machine learning, statistical analysis, modeling techniques, and database technology, data mining finds patterns and subtle relationships in data, and infers rules that allow the prediction of future results. Typical applications include market segmentation, customer profiling, fraud detection, evaluation of retail promotions, and credit risk analysis. Figure 4-2 illustrates the UML model for Any Data Mining pattern. Pattern description is provided

in the text following Figure 4-2 and the detailed pattern documentation is provided in Appendix B – Stable Design Pattern.

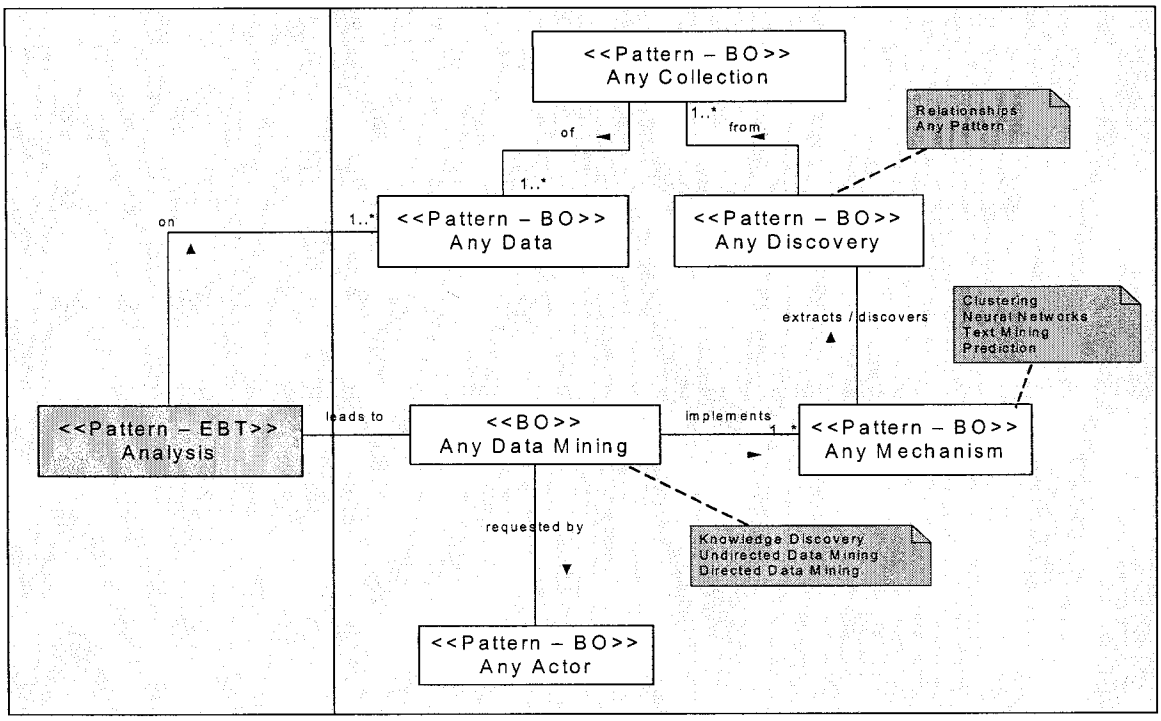


Figure 4-2 Any Data Mining Pattern (taken from Dr. Fayad's patterns archive).

4.1.2.1 Pattern Documentation

Any Data Mining is a BO, which leads to Analysis of data, so the goal of data mining is Analysis and the BO is Any Data. For example, the different types of Analysis are Exploratory Data Analysis, Market Based Analysis and Memory Based Analysis. Any Data is a part of Collection. For example, it could be from a database or from a data mart, so we have BO Any Collection. Any Discovery (a pattern or any relationship) is discovered from the Collection. Any Data Mining implements different mechanisms, such as Clustering and Neural Networks. As a result, we have the BO Any

Mechanism. Also, to represent the system or organization that requests the data mining, we have the BO Any Actor.

4.1.3 Any Data Preparation

The data that is collected and combined using data collection techniques needs some tuning. This data has to go through data cleansing, data organizing, and data translation to be in a standardized format [43]. This process is done in Any Preparation pattern.

4.1.4 Any Data Mining Mechanism

This capability considers the procedures and algorithms designed to analyze the data in databases [8]. "Any Data Mining Mechanism" provides a way to help the user select appropriate data mining algorithms, by first determining the user's actual requirements. These requirements are mapped to the appropriate goals, and then the algorithm selection is executed.

Examples of data mining mechanisms are Clustering [44 - 46], Neural Networks [47 - 50], Genetic Algorithms [51 - 55], Decision Trees [56,57], Classification Trees [17] [18,19,21,22,56], and other statistical methods [58,59].

4.1.5 Any Data Selection

This capability considers the selection process. This depends on the mapping process, i.e., the mapping between the user requirements and the goals in the system.

4.1.6 Capabilities Related to Goals

Table 4-1 lists the rest of the capabilities of the system. These capabilities are the workhorses, which help us realize the goals defined in Chapter 3.

Table 4-1 List of the Rest Capabilities

Capability	**Description of Capability**
Any Discovery	A BO that represents the discovered object or something discovered.
Any Discovery Type	Represents the different types of Discoveries. For example, knowledge discovery, star discovery, and discovery of a medicine.
Any Mechanism	Represents the mechanisms such as Research, Observation, Experiments, Clustering [44,45], Neural Networks [47-50], or Exploratory Data Analysis [60], Memory Based Analysis [1], and Market Based Analysis [1].
Any Evidence	Represents the evidence that is used as proof to validate the discovery.
Any Actor	Represents the system or user, which initiates or interacts with the patterns.
Any Data	Represents the data that is collected and the data that is analyzed.
Any Domain	Represents the available diverse domains of interest.
Any Subject	Represents the available subject areas of interest.
Any Skill	Represents an ability that can be acquired by training.
Any Aspect	Represents the different ways something can be viewed.
Any Thing	Represents the different objects, or entities, or any information, which is required to be analyzed.
Any Form	Represents the form in which the results or Any Thing appear.

CHAPTER 5 Knowledge Map and Stable Architecture Patterns

Previous sections introduced and explained Stable Analysis and Design Patterns, also called as goals and capabilities, respectively. This section will introduce the use of these goals, which are Stable Analysis Patterns [15], and capabilities, which are Stable Design Patterns, to form a valid KM. KM is a large Stable Architectural Pattern containing Stable Architecture Patterns that are generated from connecting goals and capabilities in software stability paradigm [2,3,14]. KM is important as it combines the functionality provided by EBTs and BOs and makes them work together in any given domain. KM is a big picture that demonstrates the logical flow among EBTs and BOs and it also checks the validity of the architecture by verifying the different paths in the architecture.

To generate an effective KM it is important to understand all the goals and capabilities of the topic selected. Section 5.2 discusses the development of a KM through scenarios.

5.1 Development of KM Through Scenarios

To build a correct KM some important steps need to be followed. The next few sections introduce these steps.

5.1.1 KM Through Goal Scenarios

The first step is analyzing all the goals the system should meet. Each goal has its own capabilities and properties. Some of the goal's capabilities are shared while some will belong to a specific goal. We have generated scenarios to test the commonality of some artifacts across various domains, which will become clearer after the following examples.

Let's consider the EBT Discovery. Discovery is the act of discovering something. It could be a disease, a drug, a planet, a star, or hidden patterns. Let's consider some scenarios, which will help realize some common artifacts.

Consider Discovery of dust-covered, frozen sea near the equator of Mars: The discovery, by an international team of scientists led by University College London (UCL), the Open University (OU), and the Free University of Berlin, of a frozen sea close to the equator of Mars has brought the possibility of finding life on Mars one step closer [61]. This is the first evidence of there having been recent liquid water on Mars [61]. The results of the work of the team lead by John Murray (OU), Jan-Peter Muller (UCL), and Gerhard Neukum (Free University of Berlin) were presented at an ESA Mars Science Conference at ESTEC, Netherlands on 21 February 2005 [61]. The water that formed the sea appears to have originated beneath the surface of Mars. Erupting about 5 million years ago from a series of fractures known as the Cerberus Fossae, the water flowed down in a catastrophic flood, collecting in an area 800 x 900 km and was initially an average of 45 meters deep [61]. The packed ice, which formed on the surface of the

sea, drew the attention of Mars Express scientists [61]. Table 5-1 describes the different participants of the scenario.

Table 5-1 Discovery - Scenario 1

Concept	**Description**
Any Characteristic	Pack ice before acts as the characteristic of the discovery.
Any Description	Water believed to have originated beneath the surface of Mars, and then gushed forth in a catastrophic flood after being warmed by the planet's core is the description provided.
Any Observation	Findings, experiments and calculations are the observations that lead to this discovery.
Any Result	The possibility that liquid water may still exist on Mars locked under 90 feet of ice is the result of the observations.
Any Evidence	Floes near a set of well-known fissures called the Cerberus Fossae act as evidence.
Any Period	The time period in this discovery is the time required to conduct experiments until the discovery was made.
Any Actor	University College London and scientists are the actors in this discovery.

Another scenario: Discovery of mad cow disease. Bovine Spongiform Encephalopathy (BSE) is the scientific term for the disease, which affects the brains of cattle. Soon after BSE was first discovered in the United Kingdom, it became more commonly known as "mad cow disease," most likely because of the emotional response it generated with the public. Unlike most livestock diseases, BSE is not caused by a bacterial or viral infection, but is the result of infectious prions. These are unique proteins that may bond with a cow's brain cells, altering their composition and ultimately

leading to the animal's death. Mad cow disease is transferred to cattle when they eat these infectious proteins, yet science has shown the disease can only affect those cows that are genetically susceptible [62]. Table 5-2 summarizes the participants of the scenario.

Table 5-2 Discovery - Scenario 2

Concept	**Description**
Any Evidence	Infectious prions act as the evidence.
Any Observation	Lab tests and experiments act as observations.
Any Location	United Kingdom is the location of the discovery.
Any Result	Discovery of Mad Cow disease acts as the result.
Any Actor	Scientists act as actors involved in the discovery.

Common artifacts in both scenarios are Any Discovery Mechanism, Any Evidence, Any Actor and Any Discovery. Table 5-3 describes these common artifacts and Figure 5.1 is built using these participants. A description of the pattern is provided in Chapter 3.

Table 5-3 Discovery - Final Model

Concept	**Description**
Any Discovery Type	This capability covers the different discovery types in different domains.
Any Evidence	The basis for belief or disbelief.
Any Discovery Mechanism	Method or procedure used to accomplish discovery.
Any Discovery	The discovery itself.
Any Actor	Individual or Organization involved.

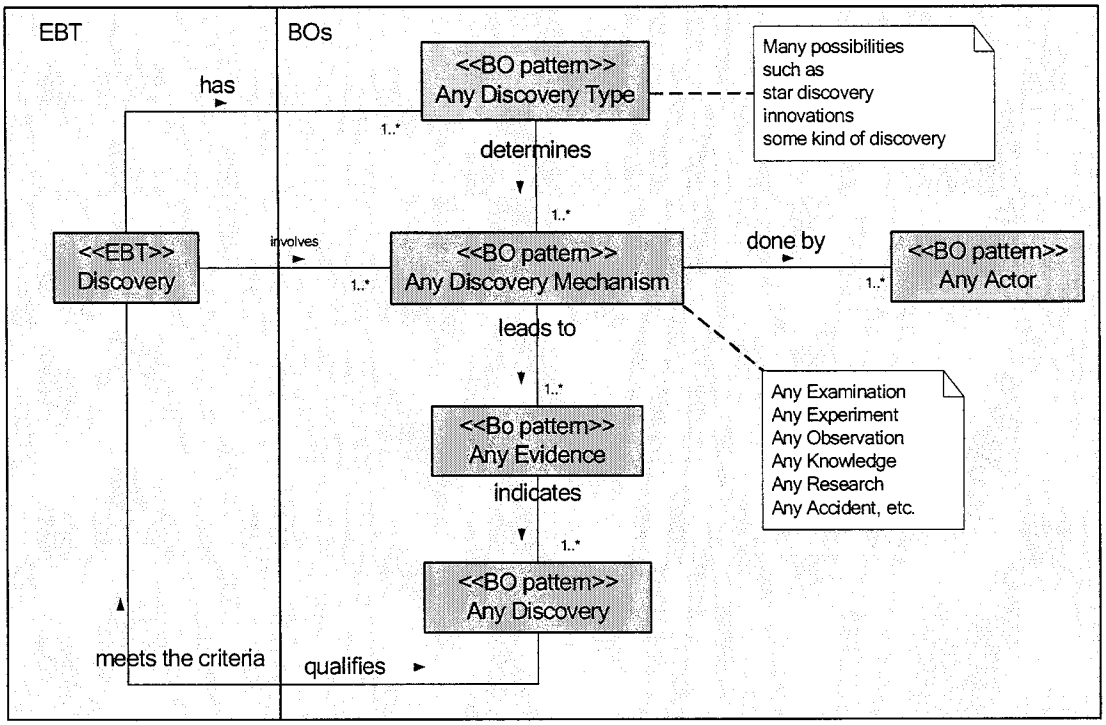


Figure 5-1 Discovery Pattern (taken from Dr. Fayad's patterns archive)

5.1.2 KM Through Capabilities

The second step is to realize the important capabilities the system should provide. Developers have to analyze the ultimate goal for each of the main capabilities. Sometimes the goals realized are new and nothing to do with the system. They still have to be considered because they help building a Stable Design Pattern from the information acquired.

The corresponding BOs [2] for the main artifact also have to be realized. The last step is to look for the relation between the goals and capabilities and map accordingly. To make this procedure clear let's consider some BOs.

Let's consider the artifact Any Data Collection. The common artifacts can be demonstrated using the "Collection in Medicine" scenario. Collection in Medicine involves gathering of urine samples, various tissues, and blood samples. These samples are gathered from different human beings and the sources of these samples are important. These samples are analyzed for different elements such as blood cells. Scientists or lab technicians analyze these samples. Description of each BO involved is given in a tabular format in Table 5-4.

Table 5-4 Any Collection - Scenario 1

Concept	**Description**
Knowledge (EBT)	Information about urine samples, blood tests, is the knowledge acquired from the tests.
Any Mechanism	Medical exams, blood tests, urine tests are the collection mechanisms.
Any Analysis	Statistical analyses of information is the analysis technique.
Any Element	Blood cells, urine samples are the elements, about which the information is collected.
Any Actor	Doctor, scientists, lab technicians are the actors involved.
Any Aspect	Any Aspect is the reason for the disease and the aid towards ailments.

Any Data Collection in Business

"Any Data Collection in Business" scenario involves gathering facts on how a process works and how a process is working from a customer's point of view. This collection is driven by knowledge of the process and guided by statistical principles. The mechanisms used to collect the information are business websites and companies.

Product managers and sales representative are responsible for the collection process.

Table 5-5 summarizes the participants in this scenario.

Table 5-5 Any Collection - Scenario 2

Concept	**Description**
Knowledge (EBT)	Facts on how a process works and how a process works from a customer's point of view is the knowledge required.
Any Mechanism	Statistical principles are the mechanisms for collecting the information.
Any Element	Facts, information, act as the data that is collected.
Any Actor	Product managers, and sales representatives are the actors involved.
Any Aspect	Better productivity, and improving profitability are the reasons for gathering knowledge.

Table 5-6 represents the common concepts from above scenarios.

Table 5-6 Any Collection - Final Model

Concept	**Description**
Knowledge (EBT)	Represents the specific information about something.
Any Mechanism	Represents the techniques and algorithms used to collect Any Element.
Any Element	Represents the object that is collected. It could be Any Data, Any Entity or Any Thing.
Any Actor	Represents Any System or Any Person, who plays an important role in Collection.
Any Aspect	Represents the different ways something can be viewed.
Any Category	Represents the different categories, which contain Any Collection and Any Element.

Figure 5-2 uses the above common artifacts to build Any Data Collection pattern. A description of the pattern is provided in Chapter 4.

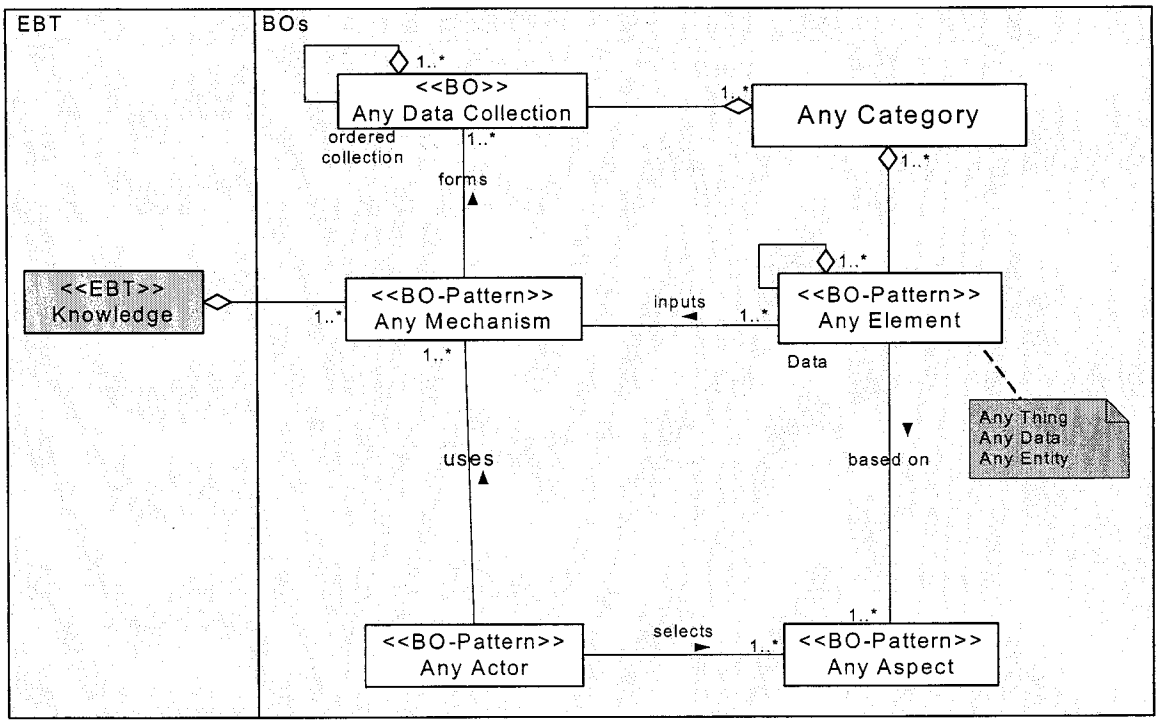


Figure 5-2 Any Collection Pattern (taken from Dr. Fayad's patterns archive)

5.2 Data Mining Goals Realized Through the Capabilities

The knowledge of data mining contains the goals and capabilities that play an important role in data mining core knowledge. It is difficult to isolate the core knowledge from the different concepts related to data mining. For example, consider data mining definitions: data mining is nontrivial extraction of implicit, previously unknown and potentially useful information from data, or the search for relationships and global patterns that exist in databases [36], and data mining is the process of analyzing data to identify patterns or relationships [36]. Data mining is a knowledge discovery and

analysis process. To successfully mine data, knowledge about the data is also very important. As a result, the goals of the system are identified. Table 5-7 represents these goals.

Table 5-7 Goals

Goals	Description
Discovery	Provided in Chapter 3.
Analysis	Provided in Chapter 3.
Knowledge	Provided in Chapter 3.

Identifying the capabilities of the system is the next step. The capabilities of the system are identified as we look at the properties the systems should provide. For example, the system should provide data collection and data preparation as properties. These capabilities are workhorses or the classes that do the actual work. Data mining capabilities deal with the artifacts of the data mining system and do not handle the other concepts related with data mining.

The goals and capabilities go hand in hand to form the KM. KM does not include Industrial Objects. It just consists of goals (EBTs) and capabilities (BOs).

5.3 Data Mining KM

Data mining KM consists of the knowledge about data mining itself. This knowledge consists of data mining goals and capabilities. These goals and capabilities are mentioned in Section 5.3.

To build a general KM first let's consider that a system has two goals or EBTs and seven capabilities of which three belong to EBT_1 and four belong to EBT_2. Then the KM for that system would look something like Figure 5-3.

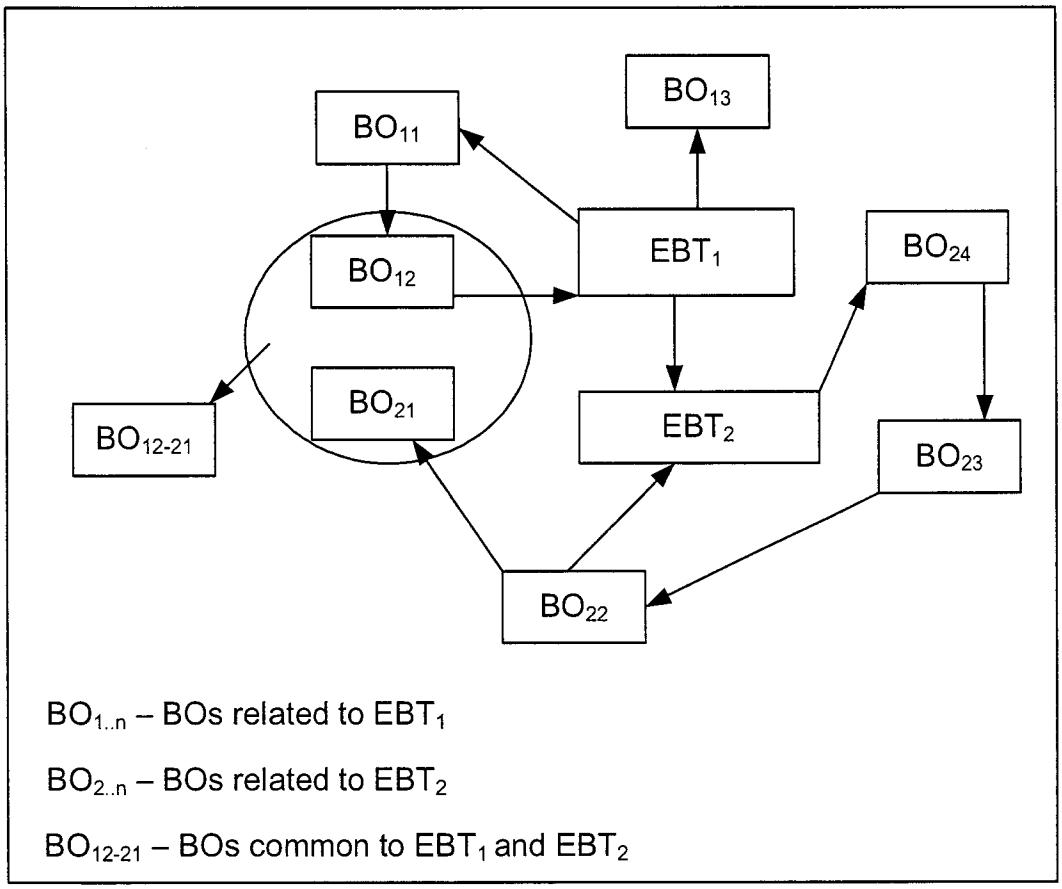


Figure 5-3 Overview of KM

Some of the facts that are important about KMs are that KMs do not have any IOs because KM forms a Stable Architectural Pattern. IO is unstable, replaceable and tangible that cannot be a part of a stable architecture. Figure 5-3 depicts that some capabilities that are common to more than one EBT are grouped together and the nomenclature of these capabilities depends on EBTs. Figure 5-4 shows a KM in data

mining domain. This KM uses the goals and capabilities that assist in accomplishing the goals specific to that domain.

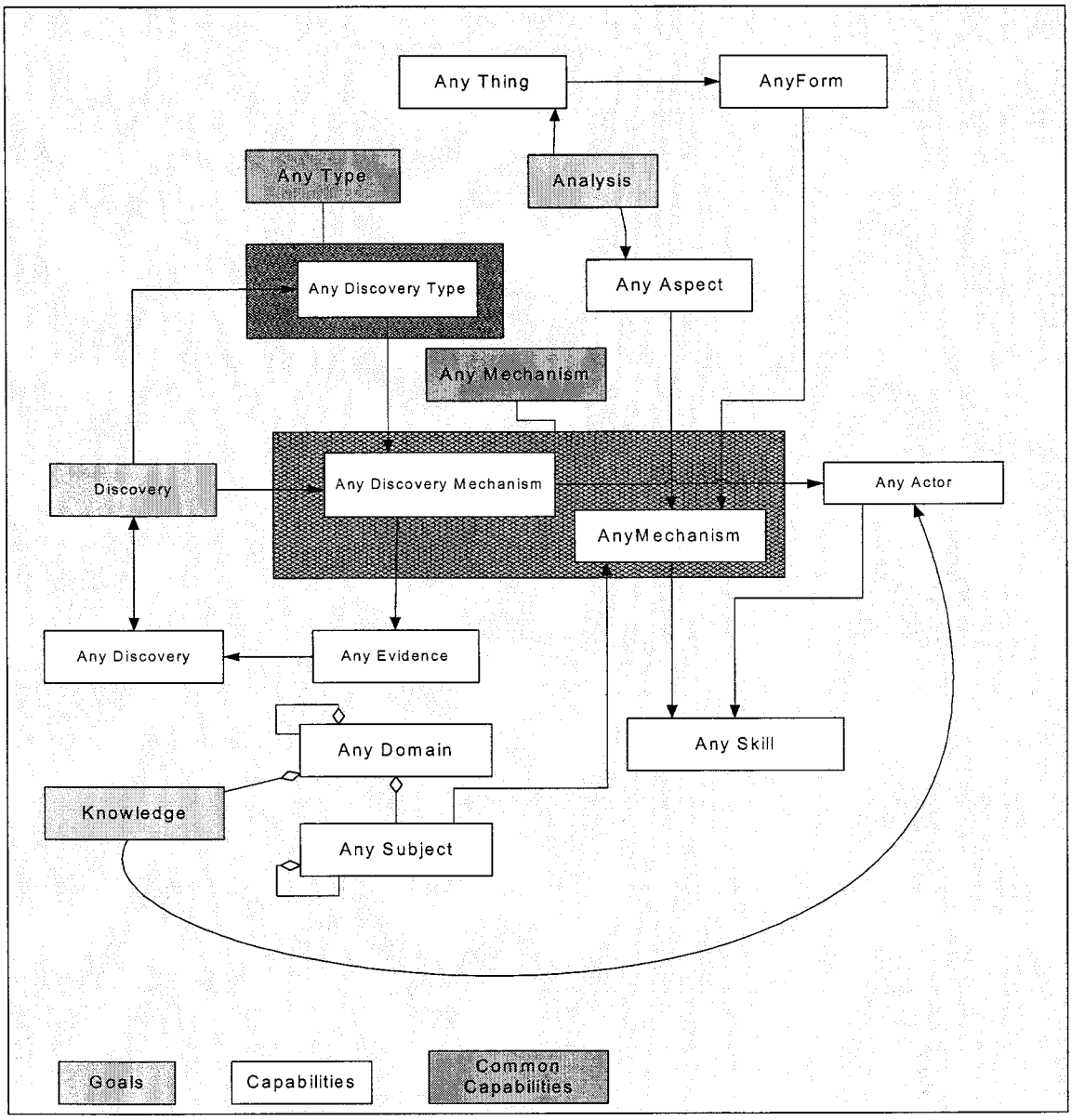


Figure 5-4 Data Mining KM

Another important aspect of KM is that it depicts the knowledge level. This knowledge level can be attained with the help of partitioning. Partitioning is explained in

the following text by providing general examples and then coming back to data mining domain. The different types of partitioning are as follow:

Partition I

The first and the easiest way of partitioning is to partition all the patterns according to their nature. An example is to partition the system into Stable Classes (EBTs and BOs), Atomic Patterns (Architectural and Design Patterns), and Architectural Patterns. All of the above patterns and classes are reusable. This is represented in Figure 5-5.

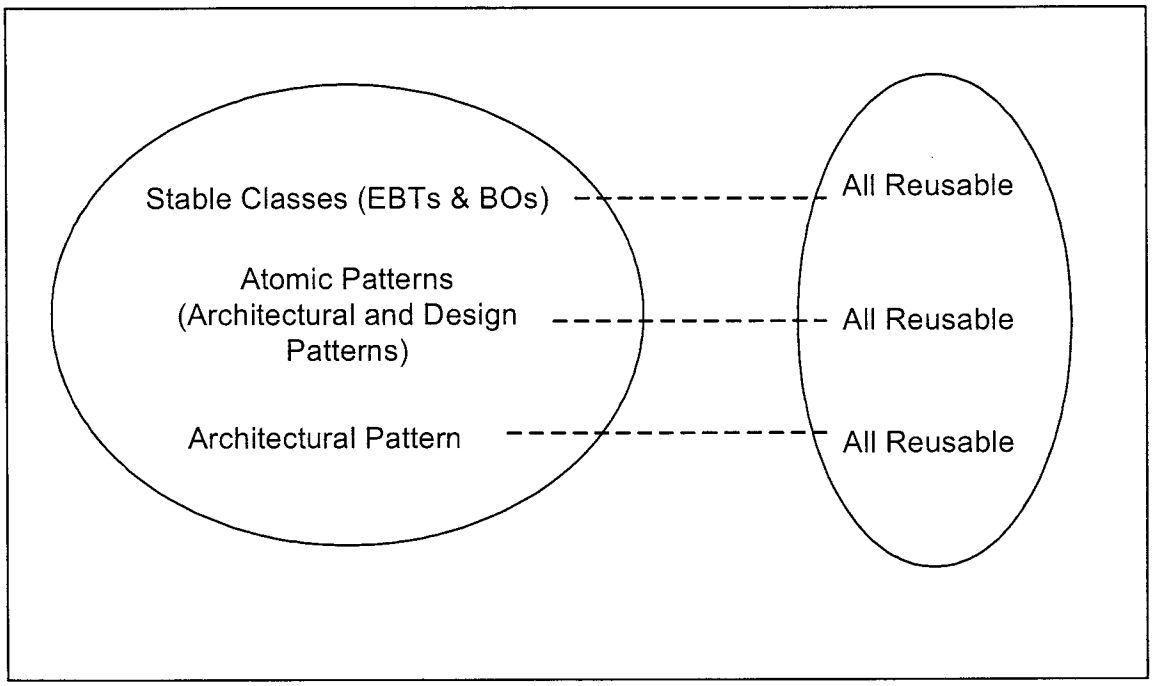


Figure 5-5 Partition I

In data mining the Stable Analysis Patterns or EBTs are Discovery, Analysis, and Knowledge. An overview of EBTs is also provided in Table 5-7. The Stable Design Patterns or BOs are Any Data Collection, Any Data Preparation, and Any Algorithm Selection.

Partition II

This type of partitioning is difficult, but important. Here we partition the system in the form of KMs. Figure 5-6 illustrates this partition. The main KM is the core KM of the system (KM1). Also, there are other KMs (KM2, KM3, and KM4) that are related to this core KM, but not a part of the core KM (KM1). All the KMs are highly reusable. Reusability means that the KM can be used again and again without any modification as a building block in different applications from the one it was originally intended for. Also, the knowledge that intersect (intersection of KMs) are reusable.

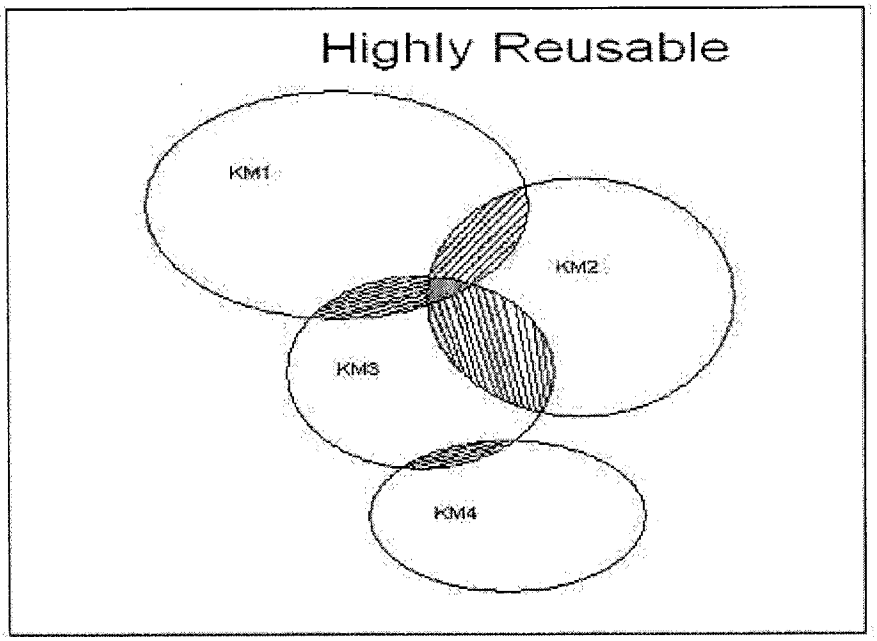


Figure 5-6 Partition II

Another significant feature of KM is that everything in KM is highly reusable. Figure 5-7 demonstrates the data mining KM with external KMs such as data

warehousing, data mart (collection of data bases), and data analysis. The common artifact in this KM is Any Data.

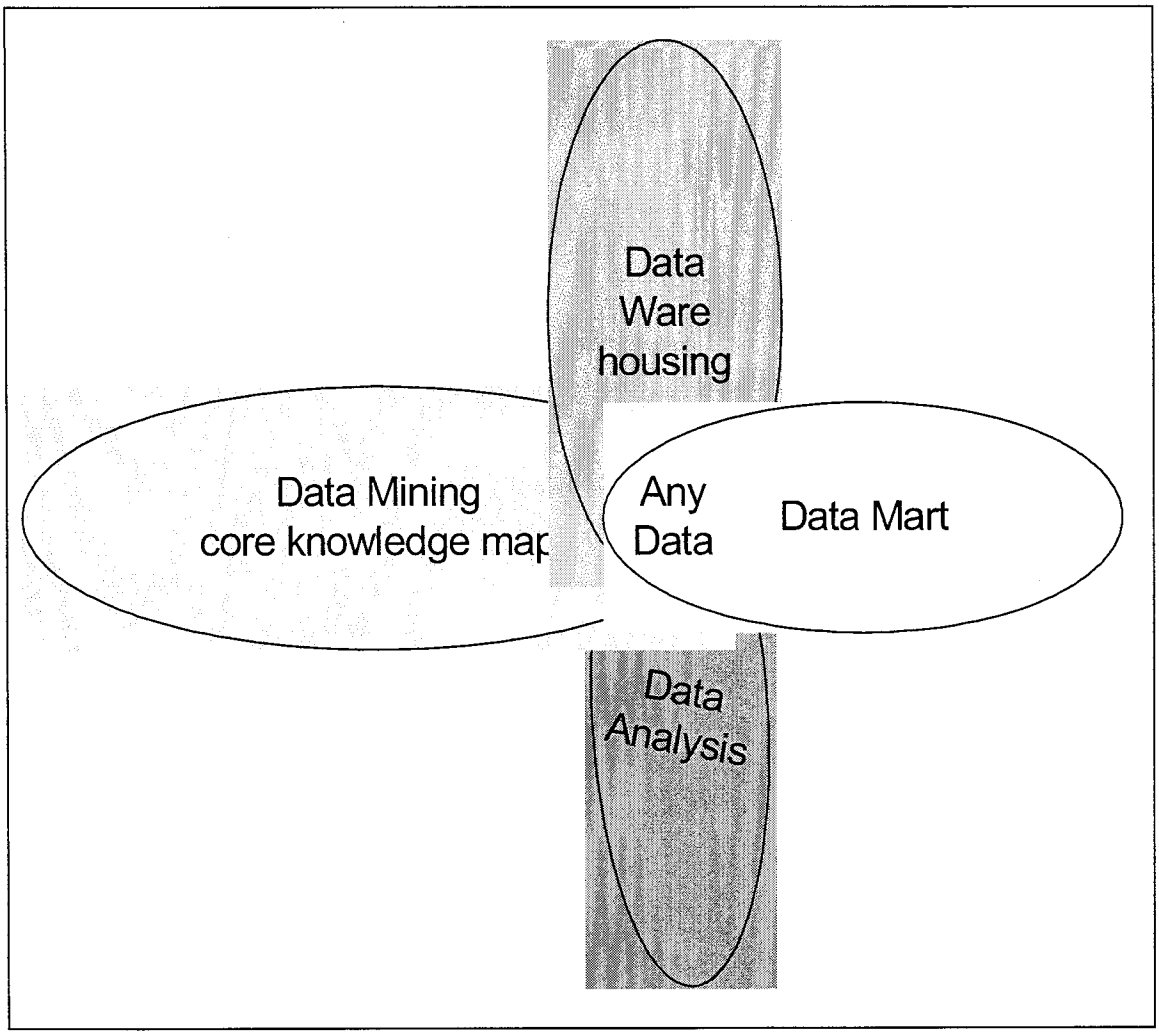


Figure 5-7 Data Mining KM with Other KMs

Partition III

This type of partition utilizes the partition II. This also adds remote KM to partition II. Figure 5-8 depicts the addition of remote KM. This remote KM is external knowledge, which may utilize some or all features of the partition II.

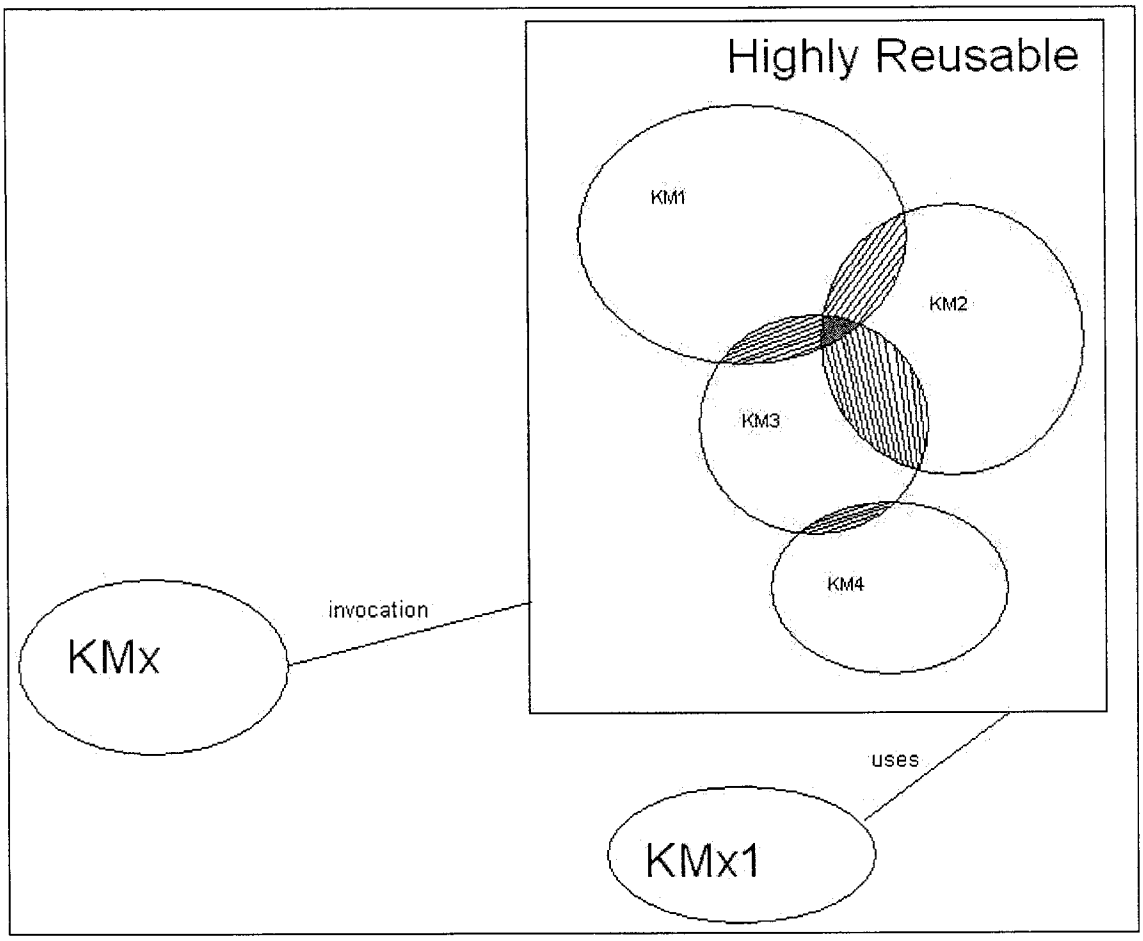


Figure 5-8 Partition III

There are many different concepts and terms related to data mining. These different concepts use data mining to discover knowledge. An example is to use data mining to discover knowledge from Data Base Management Systems. Data mining applications have the ability to query very large databases to satisfy a hypothesis ("top down" data mining); or to interrogate a database to generate new hypotheses based on rigorous statistical correlations ("bottom-up" data mining) [36]. There are many tools that incorporate data mining in correlation with other algorithms or analytical techniques to discover knowledge. Some of these techniques are Neural Networks, Clustering,

Classification, and Decision Trees. These analytical techniques are grouped together, and they interact with the goal Discovery, implementing various data mining techniques. Another concept related to data mining is Data Visualization. Data Visualization, according to Edinburgh Online Graphics Dictionary, is defined as the set of techniques used to turn a set of data into visual insight giving the data a meaningful representation by exploiting the powerful discerning capabilities of the human eye. Data is displayed as Two Dimensional (2D) or Three Dimensional (3D) images using techniques such as colorization, 3D imaging, animation, and spatial annotation creating an instant understanding from multi-variable data. Data Visualization is combined with data mining to show the final pattern or any relation discovered. Figure 5-9 illustrates the detailed picture of the data mining core knowledge with the different KMs.

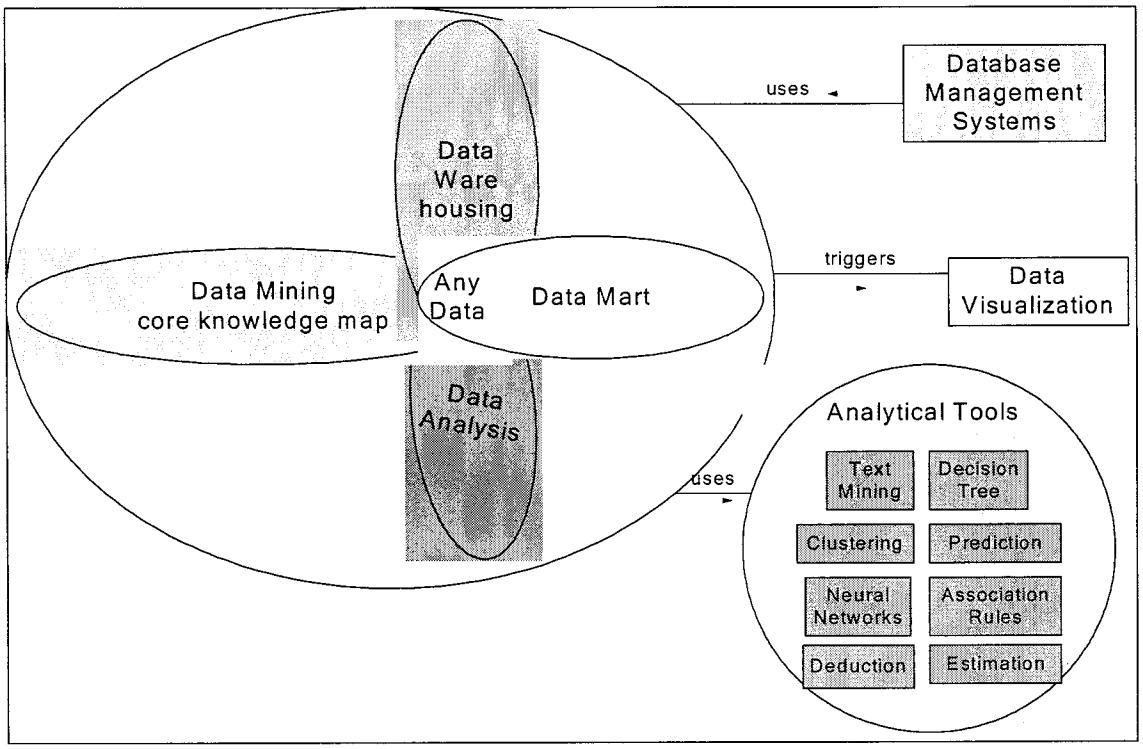


Figure 5-9 Data Mining KM with External and Remote KMs

5.4 Architecture Assessment

This section assesses the architecture with respect to different properties of the KM as stated below.

5.4.1 Properties of KM

The different properties of the KM are as follow:

5.4.1.1 Number of Architectures

The number of architectures generated from the KM reveals the extensive use of the KM and proves the reusability feature of the KM. For example, architectures involving only Discovery and Analysis individually can be formed. On the same count,

architectures involving Discovery and Analysis can be formed, which is shown in Figure 5-10. Also, with the use of external KMs, architecture concerning Data Visualization or any analytical tools, or Database Management Systems can be formed.

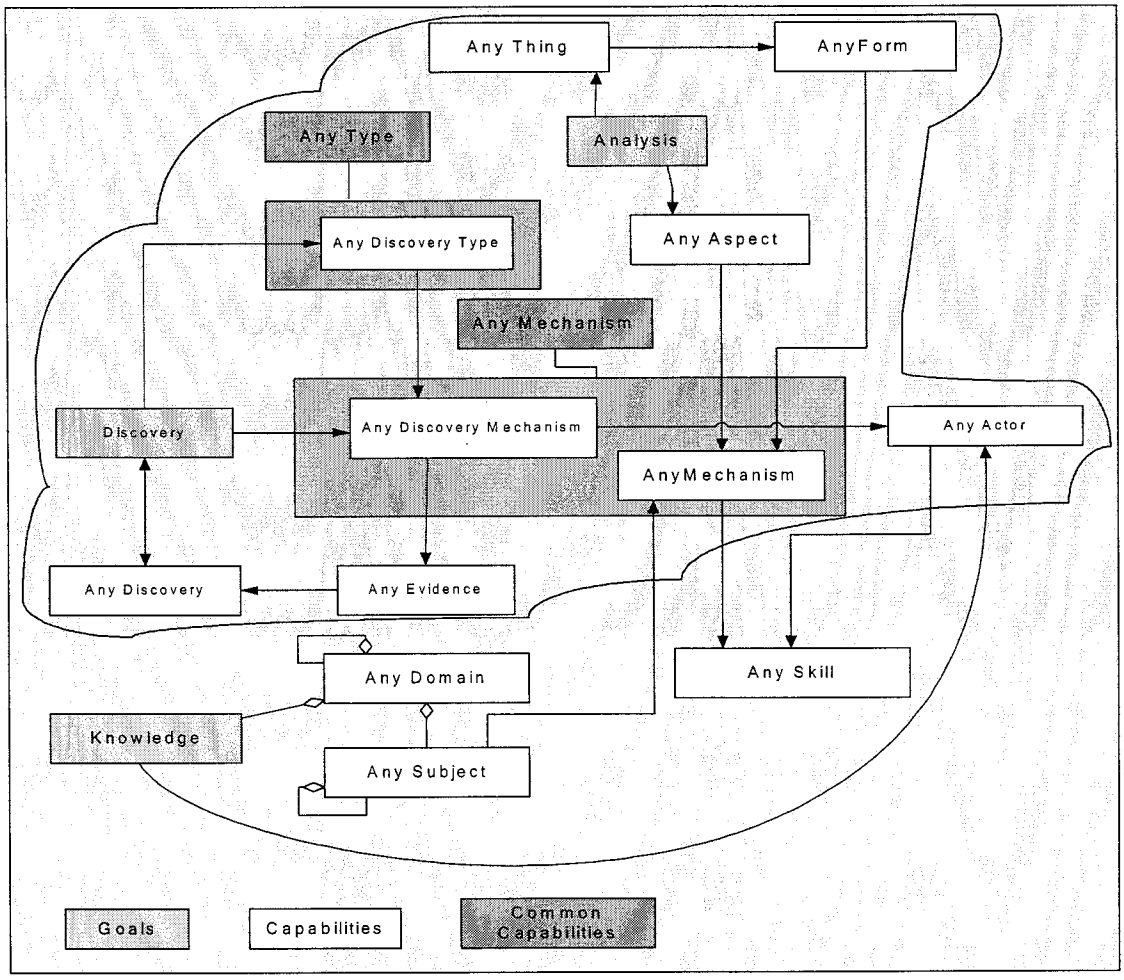


Figure 5-10 Example of Architecture Generated Using Discovery and Analysis

5.4.1.2 Direction

The generated architectures from KM have to be qualified and proven. The direction between the EBTs and BOs qualifies the architecture. For example, architectures involving any analytical algorithms such as Clustering have to be qualified.

Clustering from the external KM is connected to Any Mechanism. When data mining application requires implementation of Clustering or any other analytical tools, the direction is as follows:

1. Discovery → Any Discovery Type → Any Mechanism → Any Evidence → Any Discovery
2. Analysis → Any Aspect → Any Mechanism → Any Skill → Any Actor

5.4.1.3 Number of Applications

The KM also gives us an idea of the number of applications that can be built on top of it. An unlimited number of applications can be built on top of KM. Using the data mining core KM along with external KM and remote KM any number of applications can be built.

5.4.1.4 Return on Investment

The KM also demonstrates the ROI on the system. ROI is defined as a figure of merit used to help make capital investment decisions. ROI is calculated by considering the annual benefit divided by the investment amount [36]. The extensive use and reuse of KM demonstrates the ROI. Since the KM is extremely usable and reusable, the ROI is huge. Since unlimited number of applications can be built on KM, the ROI is enormous.

5.4.1.5 Out of Scope

Out of Scope is an important property of KM. Developers should prevent going out of scope of the core KM as it distracts from the main objective. For example, the core KM of data mining contains goals and capabilities related to data mining such as Discovery, Analysis and Knowledge. It contains capabilities related to these goals such

as Any Discovery Type, and Any Discovery Mechanism. Similarly, knowledge related to Data Visualization or Database Management Systems lie out of scope of data mining even though they use the functionality of data mining core knowledge.

5.4.1.6 Rearrangement

The goals and the capabilities in the KM can be rearranged to demonstrate validity of the architectures and the KMs that are generated. This rearrangement should not affect the functionality of KM.

CHAPTER 6 Data Mining Development Scenarios and Implementation Details

This chapter provides detailed specifications for all the models provided in the previous chapters and implementation details for the models put into operation. Two patterns have been implemented namely Discovery and Any Data Mining pattern. Tables 6-1, 6-2, 6-3, 6-4, and 6-5 briefly show specifications of five models, namely Discovery, Knowledge, Analysis, Any Data Collection, and Any Data Mining.

Table 6-1 Specification of EBT Discovery

EBT	**BO**
Discovery	Any Discovery Type
	Any Discovery Mechanism
	Any Evidence
	Any Actor
	Any Discovery

Table 6-2 Specification of EBT Knowledge

EBT	**BO**
Knowledge	Any Domain
	Any Mechanism
	Any Subject
	Any Skill
	Any Actor

Table 6-3 Specification of EBT Analysis

EBT	**BO**
Analysis	Any Thing
	Any Aspect
	Any Mechanism
	Any Form

Table 6-4 Specification of BO Any Data Preparation

EBT	**BO**	**BOs**
Knowledge	Any Data Collection	Any Actor
		Any Mechanism
		Any Aspect
		Any Element
		Any Category

Table 6-5 Specification of BO Any Data Collection

EBT	**BO**	**BOs**
Analysis	Any Data Mining	Any Data
		Any Collection
		Any Mechanism
		Any Actor
		Any Discovery

6.1 Type Oriented Paradigm

In this section, Type Oriented Paradigm and Type versus Classes are described. Type Oriented Paradigm falls in two categories: Inheritance and Containment.

6.1.1 Inheritance

Inheritance describes the parent-child or hierarchical relationship between classes.

6.1.1.1 Type Names an Interface – Example 1

This characteristic is described in the form of a Type with the corresponding Interface in Table 6-6 and then this example is used as a reference in the data mining scenario. In this example the type is Integer and the Interfaces are add, subtract, multiply, divide, mod, and other integer operations. Similarly, for Type Stack the Interfaces are pop, push, length, full, and empty.

Table 6-6 Example 1 of Type Names an Interface

Type	**Interface**
Integer	+, -, /, %, *
Stack	pop(), push(), length(), full(), empty()

6.1.1.2 Class Implements a Type – Example 1

To describe this property, an example of Stack with the corresponding sub classes and Interfaces is shown in Table 6-7 and Figure 6-1. The subclasses of Stack type are Arrays, Single Linked Lists, and Hash Tables.

Table 6-7 Example 1 of Class Implements a Type

Super class	Subclass
Stack	ARYs
	SLLs
	HTs

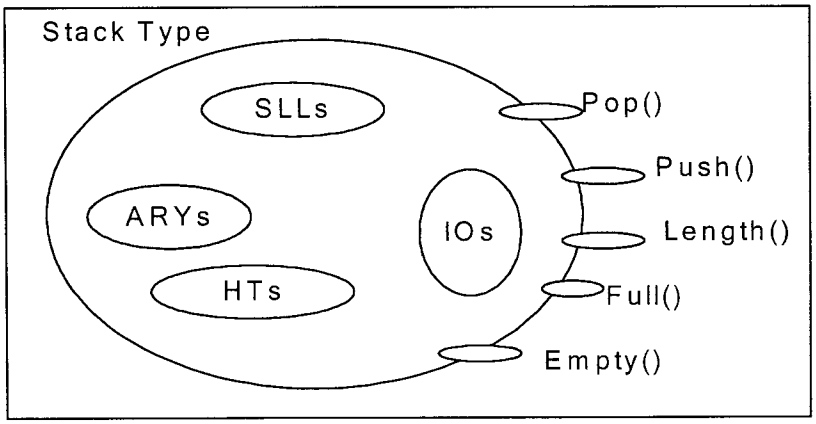


Figure 6-1 Representation of Type, Class, and Interfaces in Inheritance

6.1.1.3 Type Names an Interface - Data Mining Domain

Using the above examples as reference Table 6-8 shows the different Types and their corresponding Interfaces.

Table 6-8 Type Names an Interface – Collection

Type	Interface
Collection	collect()
	grow()
	shrink()
	size()

6.1.1.4 Class Implements a Type – Data Mining Domain

Table 6-9 and Figure 6-2 represent "Class implements a Type" in collection type.

Table 6-9 Class Iimplements a Type – Collection

Super class	**Subclass**
Collection	Library
	Credit collection
	Stamp Collection
	Dictionary

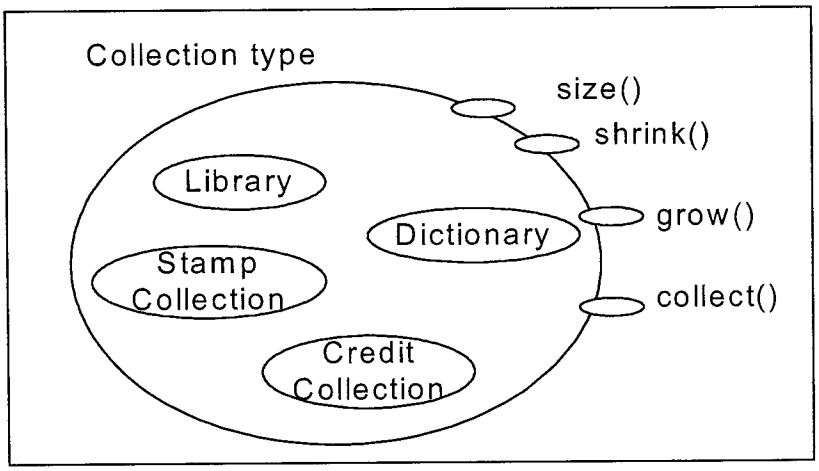


Figure 6-2 Representation of Type, Class, and Interfaces in Collection

6.1.2 Containment

Containment represents the part-whole or aggregation relationship between classes.

6.1.2.1 Type Names an Interface – Example 1

Example in the form of Any Media and its Interfaces is shown in Table 6-10.

Table 6-10 Example 2 of Type Names an Interface

Type	Interface
Media	open()
	close()
	facilitate()
	provide()

6.1.2.2 Class Implements a Type – Example 1

Table 6-11 and Figure 6-3 show Any Media and its contained classes. This example represents an aggregation relationship between the whole and part classes. Because the relationship is aggregation the part classes have their own behavior apart from whole classes. For example, Scanner has operation scan(), and Printer has operation print(), but they are a part of class Any Media.

Table 6-11 Example 2 of Class Implements a Type Any Media

Whole	Parts
Media	Scanner
	Printer
	TV
	Radio
	Computer

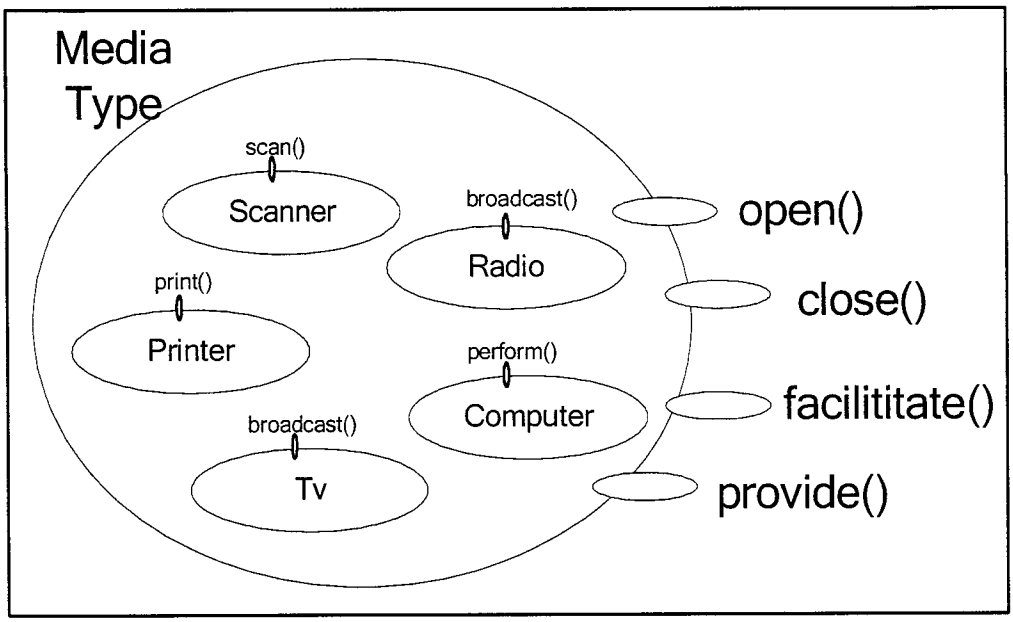


Figure 6-3 Example 1 Representation of Type, Class and Interfaces

6.1.2.3 Type Names an Interface – Data Mining Domain

Using the above examples as reference Table 6-12 shows the different Types with their corresponding Interfaces.

Table 6-12 Type Names an Interface - Mechanism

Type	**Interface**
Mechanism	implement()
	provideResults()

6.1.2.4 Class Implements a Type - Data Mining Domain

Table 6-13 and Figure 6-4 represent the "Class implements a Type" in Mechanism Type.

Table 6-13 Class Implements a Type - Mechanism

Whole	Parts
Mechanism	Clustering
	Text Mining
	Neural Networks
	Genetic Algorithms
	Research
	Observation
	Experiments

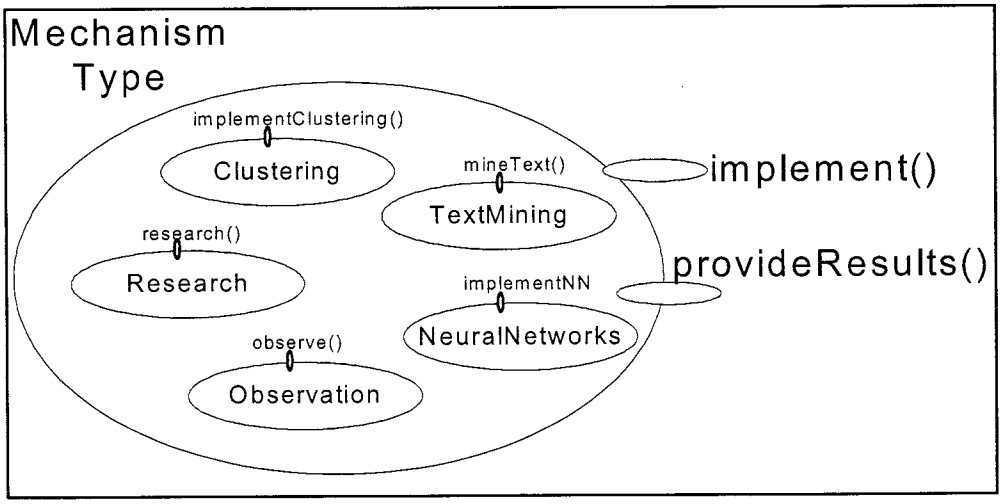


Figure 6-4 Representation of Type, Class and Interfaces - Collection

6.2 Hook

Hooks act as extension points to help attach the peripheral replaceable Industrial Objects to Stable Design Patterns also termed as BOs. Hook customizes the KM to adapt to applications of interest. In data mining, hooks facilitate the use of core knowledge across different domains with different applications. For example, the Discovery pattern

can be used in an application for the discovery of a planet and at the same time it can be used in knowledge discovery application. Also, Any Data Mining pattern can be used in credit card fraud detection application and moviegoer's application where the data mining mechanisms are different in either application. To demonstrate the usage of hooks, Table 6-14, and Table 6-15 provide an example of interaction between hooks and BOs and IOs in knowledge discovery application and planetary system with respect to Discovery pattern.

Table 6-14 Example of Hooks Used in Knowledge Discovery Application

EBT	**BOs**	**Hooks**	**IOs**
Discovery	AnyDiscovery	AnyDiscoveryHook	Pattern
	AnyDiscoveryType	AnyDiscoveryTypeHook	Knowledge Discovery
	AnyDiscoveryMechanism	AnyMechanismHook	Clustering, TextMining
	AnyEvidence	AnyEvidenceHook	Facts
	AnyActor	AnyActorHook	Scientist

Table 6-15 Example of Hooks Used in Planetary System Application

EBT	**BOs**	**Hooks**	**IOs**
Discovery	AnyDiscovery	AnyDiscoveryHook1	Planet
	AnyDiscoveryType	AnyDiscoveryTypeHook1	Planetary System
	AnyDiscoveryMechanism	AnyMechanismHook1	Research
	AnyEvidence	AnyEvidenceHook1	Observation
	AnyActor	AnyActorHook1	Scientist

6.2.1 Inheritance Hook

To demonstrate usability of hooks consider an AnyActor hook, which is an inheritance hook as shown in Table 6-16.

Table 6-16 Hook Template for Inheritance AnyActorHook

Name	AnyActorHook
Requirements	AnyActorHook inheritance Hook
Type or Level	Enabling a feature
Area	None
Uses	None
Participants	AnyActor, Scientist
Changes	<Changes> <Statement>NEW SUBCLASS Scientist OF AnyActor</Statement> <Statement>Scientist.implementMechanism() OVERRIDES AnyActor.implementMechanism() </Statement> </Changes>
Constraints	None
Comments	AnyActorHook is an inheritance Hook.

6.2.2 Aggregation Hook

To demonstrate the usage of aggregation hook let's consider an example of AnyMechanism hook as shown in Table 6-17.

Table 6-17 Hook Template for Aggregation AnyMechanismHook

Name	AnyMechanismHook
Requirements	AnyMechanismHook is a aggregation Hook
Type or Level	Adding a feature
Area	None
Uses	None
Participants	AnyMechanism, Clustering, TextMining
Changes	<Changes> <Statement>AnyMechanism ATTACH Clustering</Statement> <Statement>AnyMechanism ATTACH TextMining</Statement> <Statement>Clustering.implementMechanism OVERRIDES AnyMechanism.implementMechanism() </Statement> <Statement>TextMining.implementMechanism OVERRIDES AnyMechanism.implementMechanism() </Statement> </Changes>
Constraints	None
Comments	AnyMechanismHook is an aggregation Hook.

6.3 Model Based Architecture

The most recent innovations [63] have focused on notations and tools that allow users to express system perspectives of value to software architects and developers in ways that are readily mapped into the programming language code that can be compiled for a particular operating system platform. The current state of this practice employs the Unified Modeling Language (UML) as the primary modeling notation [64]. "The UML allows development teams to capture a variety of important characteristics of a system in corresponding models. Transformations among these models are primarily manual. UML modeling tools typically support requirements traceability and dependency relationships among modeling elements, with supporting documents and complementary consulting offerings providing best practice guidance on how to maintain synchronized models as part of a large-scale development effort" [64]. This is well demonstrated in Figure 6-5

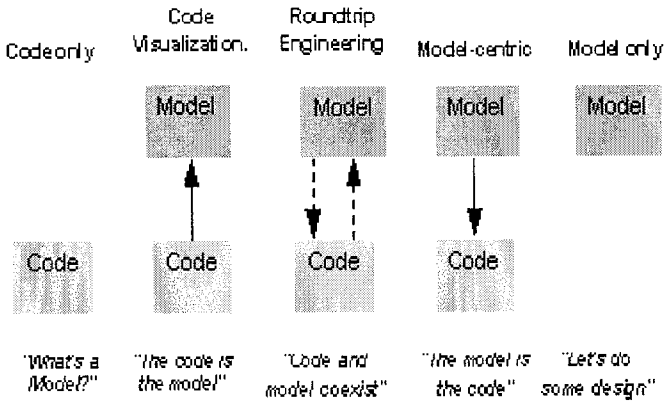


Figure 6-5 Model Driven Architecture [20]

6.3.1 Data Mining Model Based Architecture

Figure 6-6 represents the data mining model based architecture. These concepts (patterns) are explained more in the next section.

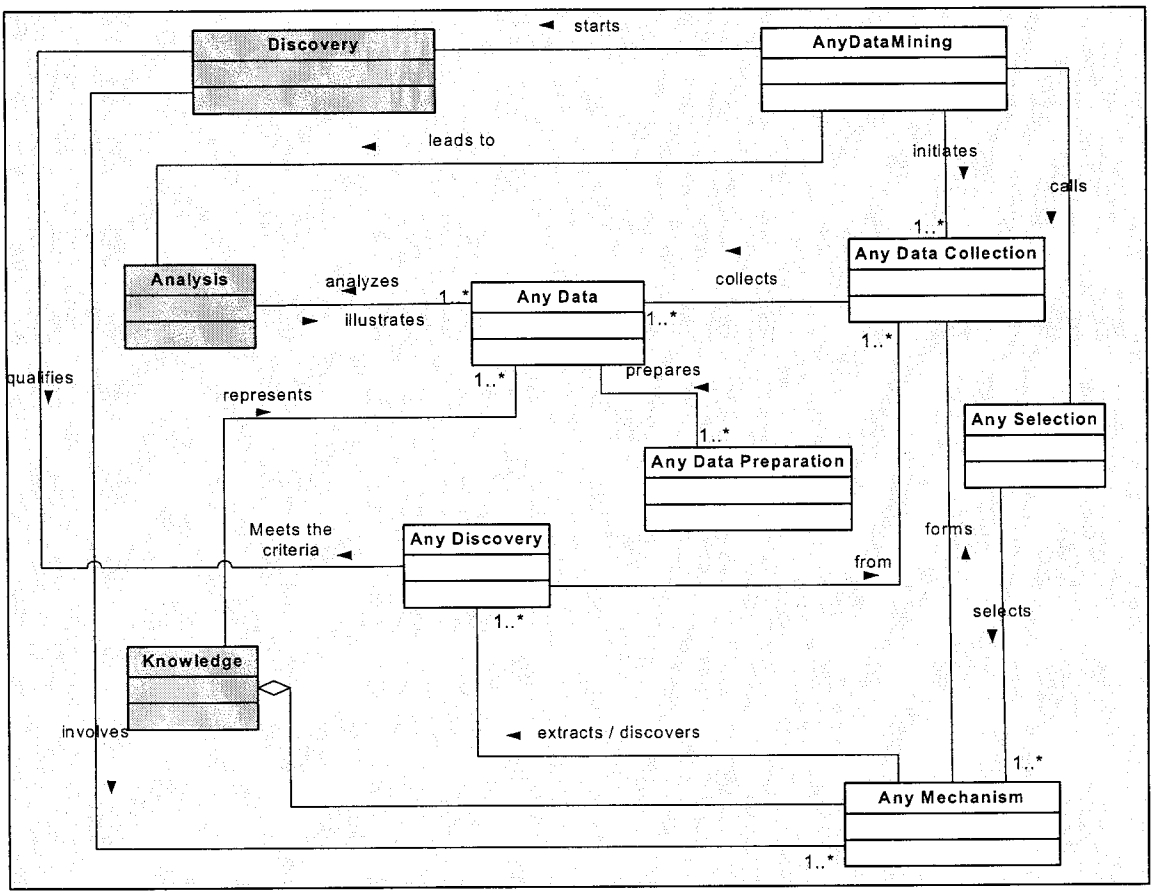


Figure 6-6 Data Mining Model Based Architecture

6.3.1.1 Description of Data Mining Model Based Architecture

Data mining model based architecture connects all goals and capabilities mentioned in Chapter 3 and Chapter 4. Discovery is an EBT [14] for data mining application. The goal of a data mining application is to discover patterns in data to fulfill the user-specific goals. Therefore, the Discovery EBT pattern will aid the application to fulfill its goals. Analysis is a process of systematically applying statistical and logical

techniques to describe, summarize, and compare data. Knowledge is awareness and understanding of facts, truth, or information gained in the form of experience or learning [24]. Knowledge is also defined as "information combined with experience, context, interpretation, and reflection. It is a high-value form of information that is ready to apply to decisions and actions [24]."

Any Data Mining is a BO, which is an information extraction activity whose goal is to discover hidden facts contained in databases [42]. Using a combination of machine learning, statistical analysis, modeling techniques, and database technology, data mining finds patterns and subtle relationships in data and infers rules that allow the prediction of future results [65]. Any Data Collection is a BO, which consists of collections of data, procedures, or methods to be executed and processed to develop the methodology for any domain. These are also responsible for organizing the collection of things it contains. Any Data Preparation is also a BO, which is responsible for fine-tuning the information for integration with the next phase. Any Mechanism is a BO, which is used to represent a technique or an algorithm. A technique can in turn invoke other techniques. Any Data, another BO, represents data or output results, generated by a procedure with a standard format of representation. Any Algorithm Selection is a BO listing a choice of algorithms and executes the choice selected. This is the basic selection process. The process of the logical flow shown in the Figure 6-6 is as follows:

Any data mining application invokes Any Data Mining pattern. Any Data Mining pattern uses Discovery pattern, which implements one to many Mechanism patterns to discover hidden trends or information. Any Data Mining pattern leads to Analysis of

data. Data mining process also requires Any Data Collection pattern and Any Data Preparation pattern to perform its functionalities. Any Data Collection and Any Data Preparation patterns work on Any Data pattern to transform the data. Data mining also uses Any Algorithm Selection pattern to select the algorithm according to the classified goals of the system. Any Algorithm Selection pattern also invokes one to many mechanisms, which perform the selection and represent the algorithms.

6.3.1.2 Description of Credit Card Fraud Detection Application

Figure 6-6 represented the general model based architecture for data mining. Figure 6-7 uses data mining model based architecture shown in Figure 6-6 in credit card fraud detection application. Following is the description of Figure 6-7 and the classes involved.

Credit card fraud detection application (CCFraudDetection) is the main class, which invokes other classes in the application. It presents the initial UI screen to the user asking for inputs, including the goals of the user. It collects the data accumulated and organized by the DataMart class. DataMart class is responsible for preliminary data acquisition, recording the lists of data, location of data, methods used for acquisition, and problems and solutions in preliminary acquisition. Sorting is a function responsible for sorting the data in ascending or descending order. RawData class is responsible for storing and sorting the data. Format Translator class translates raw data into a standard form in Data Mining algorithms. Format Translator class also does syntactic modifications to the data without changing its meaning, including reordering of the attributes and changes related to the constraints of the modeling tools such as removing

commas or tabs, or replacing special characters with new sets of special characters. TranslatedData class represents the data that is generated by the Format Translator class, translated and prepared according to requirements of the specific algorithms. CCFraudDetection application class invokes GoalClassifier class after the completion of data preparation, translating the goals entered by the user into data mining outputs using translation rules specific to each application. AlgorithmSelector class selects the algorithm(s) matching the data mining goals determined in GoalClassifier class. Clustering is one of the analytical techniques. These techniques are used to work on the data and produce the required results or patterns. Figure 6-7 represents the data mining model based architecture in credit card fraud detection application.

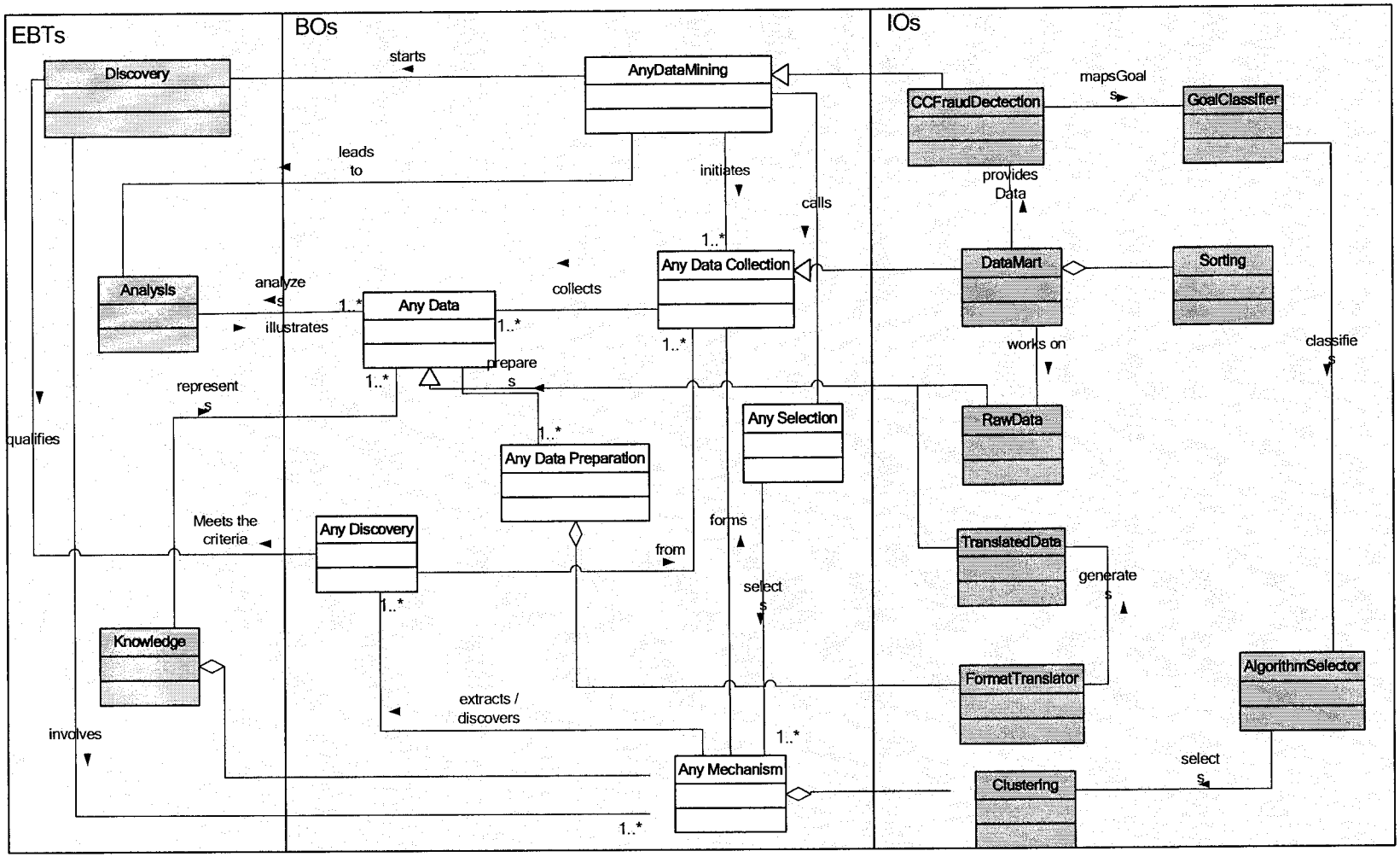


Figure 6-7 Model Based Architecture in Credit Card Fraud Detection Application

6.3.1.3 Description of Knowledge Discovery Application

Figure 6-8 represents the data mining model based architecture in knowledge discovery application. The description of Figure 6-8 and the classes are as follow.

KnowledgeDiscovery class is the main class of the application, which invokes other classes. It presents the UI screen to the user asking for goals of the user. Survey class is responsible for preliminary data collection. It records the list of data, location of data, and the contents. Organizer class is responsible for organizing the data. This primarily includes defining the volume of data (number of examples and attributes), identities and meanings of individual attributes, and description of the initial format of the data. TreatData class checks the completeness and correctness of data including the consistency of individual attribute values and types, and quantity and distribution of missing values. It creates a new database of raw data for further analysis.

TuneData class optimizes and cleans data by using the following techniques: data normalization, data smoothing, treatment of missing values, and data reduction.

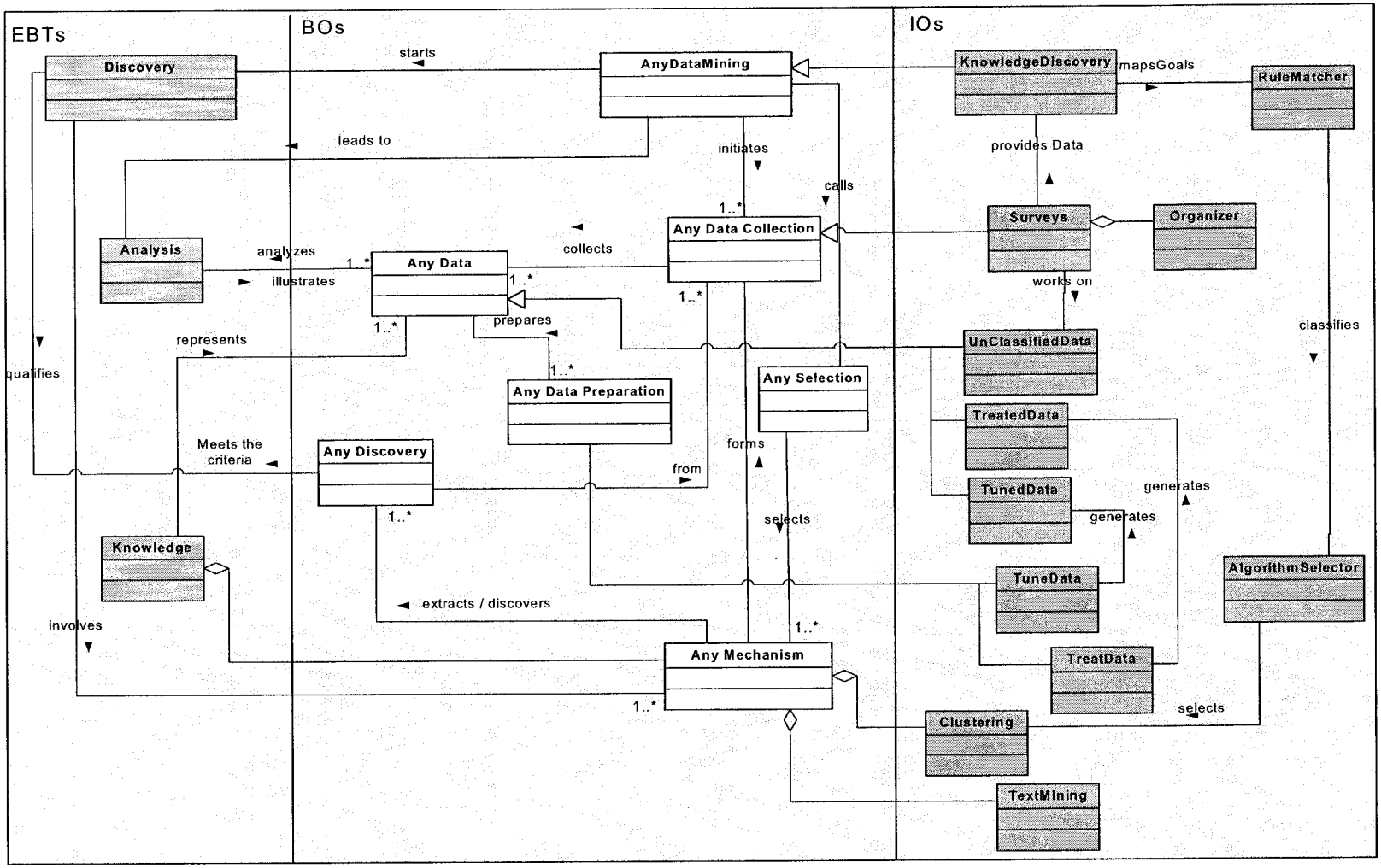


Figure 6-8 Model Based Architecture in Knowledge Discovery Application.

CHAPTER 7 Data Mining Deployment

Deployment is defined as an activity in which an application is delivered to a customer organization and introduced to the user organization [8]. Deployment is also defined as the process of installing an application, service, or content on to one or more computer systems. The data mining applications of this thesis have been developed using Core Java [66], Java Server Pages (JSPs) [66], Servlets [66], and JDBC [66]. To demonstrate ease of use and fast deployment qualities, it is important for the application or the tool to be deployed in short span of time without much effort. It is also important to address the quality factors during deployment. These factors are those responsible for the assessment of the product in terms of efficiency of use, task adequateness, cognitive workload, robustness, learning cost, and user acceptance. Quality factors are the result of the decomposition of the term 'quality of the application.' These are variables, which reflect independent quality aspects of the application and validation questions in order to allow meaningful measurement.

7.1 Deployment of Data Mining Patterns in Different Application Domains

The Data Mining pattern and the Discovery pattern can be applied across different domains. To demonstrate the reusability feature of Stable Architectural Patterns, different applications have been assessed in different domains. These applications have been described in detail in Chapter 6. Figures 7-1 and 7-2 briefly summarize the applications.

KnowledgeDiscovery class invokes RuleMatcher class after the completion of data preparation. It translates the rules entered by the user into data mining goals using translation rules specific to the application. AlgorithmSelector class selects the algorithm(s) matching the data mining goals determined in RuleMatcher class. Clustering and Text Mining are the analytical techniques, which are used to modify the data and produce the required results or patterns. UnClassifiedData represents the raw data that the surveys have collected and the organizer is working on. TreatedData class represents the data that is generated by the TreatData class after treating the data and validating it for its correctness. TunedData class represents the data generated by TuneData class after cleaning and tuning the data.

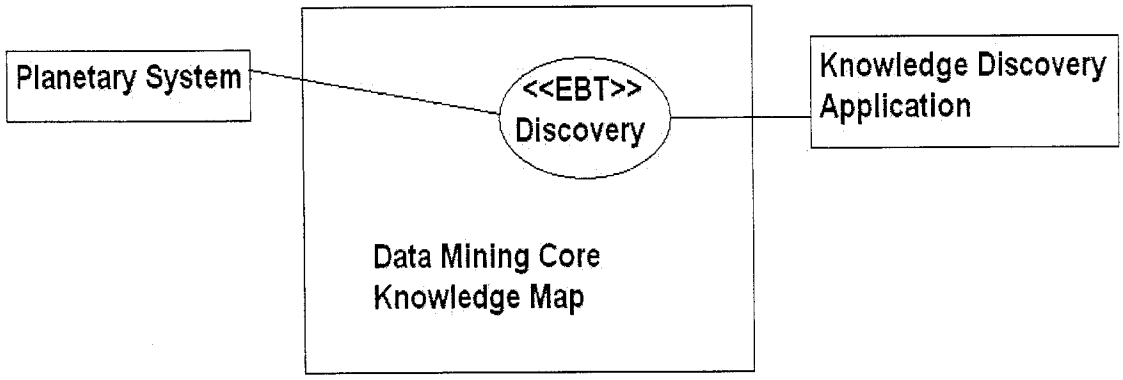


Figure 7-1 Overview of Discovery Pattern and its Applications

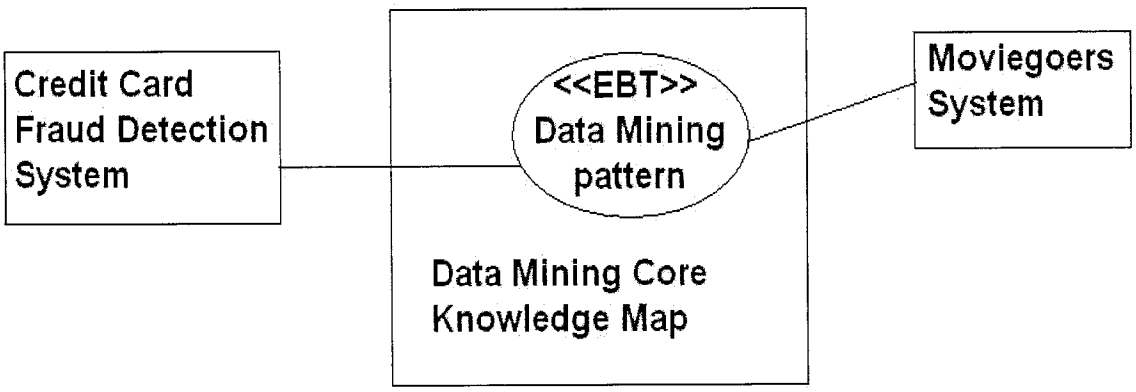


Figure 7-2 Overview of Data Mining Pattern and its Applications

The applications mentioned above are deployed using .war files [66]. These applications are deployed across an Apache Tomcat web server [67]. Applications in the form of .war files are easy to deploy. In addition, the JDK tool kit [66] and Apache Tomcat [67] web server is needed. The web server automatically installs the .war files once Tomcat is started.

7.1.1 Quality Factors

There are many quality factors that are associated with the core data mining KM. Some of them are as follow.

7.1.1.1 Accuracy

The data mining core KM satisfies this quality factor. When applied to data, Accuracy refers to the rate of correct values in the data. When applied to models, Accuracy refers to the degree of fit between the model and the data. This measures how error-free the model's predictions are. The applications developed on top of the KM demonstrate the accuracy of the models and the accuracy of the KM.

7.1.1.2 Scalability

The Stable Analysis Pattern Discovery and Stable Design Patterns Any Data Mining, Any Collection, and Any Preparation are modeled such that they can be applied across different domains. Different applications can also use these models; as a result, the models and KM as a whole are scalable.

7.1.1.3 Robustness

The patterns are modeled using SSM and SPL. As a result, the patterns and the KM are stable, reliable, and valid. For example, the Discovery Stable Analysis Pattern is modeled such that it incorporates all the available discovery mechanisms and allows adding any new discovery mechanisms. Other patterns are also modeled such that they are stable in nature.

7.1.1.4 Simplicity

The patterns and the KM are simple in nature and easy to comprehend. Figure 7-3 describes the interaction of KM with the quality factors.

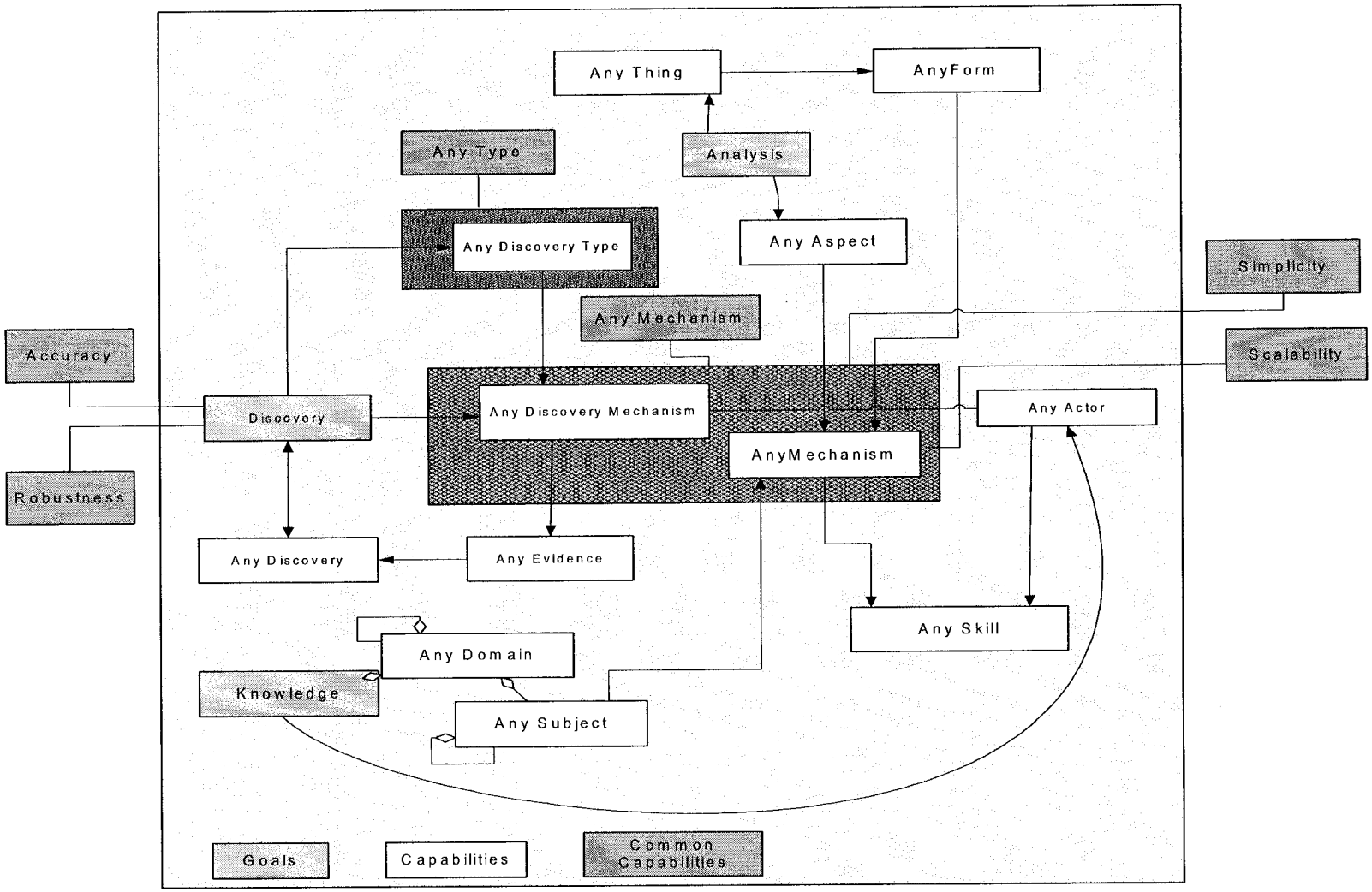


Figure 7-3 KM with Quality Factors

7.2 Validations and Verification

Validation and verification are means for testing the models and the KM. When Stable Analysis Patterns, such as Discovery and Knowledge were modeled, their respective capabilities were chosen and tested across different applications and domains to verify their validity. When Stable Design Patterns, such as Any Data Collection, Any Data Preparation, and Any Data Mining were modeled, their respective goals were tested across different domains. This verification and validation is done using stable patterns and software stability. For example, let's consider Discovery pattern in two different scenarios.

Scenario 1

This scenario demonstrates the applicability of the Discovery pattern in the discovery of Vitamin K [68]. Vitamin K discovery is a part of Any Discovery Type pattern. The individual discovery mechanisms are Experiments, Investigation and Research. The results generated after implementing these mechanisms act as evidence and Vitamin K (Any Discovery) is discovered. The scientists involved in this discovery process are responsible for initiating the discovery process, conducting the Research, and Experiments. Scientist in this scenario plays the role of an actor (Any Actor).

Scenario 2

This scenario demonstrates the applicability of the Discovery pattern in a research application to track information gathering in a particular field of technology (data mining). Knowledge discovery [10,11] is a part of Any Discovery Type pattern. The different discovery mechanisms are the different data mining algorithms such as

Clustering [44 - 46], Neural Networks [47- 50], Classification, Regression Trees, and Decision Trees [56,57].

These two scenarios demonstrate the applicability of Discovery Stable Analysis Pattern in two different domains and applications. Similarly, the patterns for other Stable Analysis and Design Patterns are modeled. A re-check of these patterns is made to assure applicability in each domain.

CHAPTER 8 Conclusions

Data mining is important in today's world to improve productivity, planning, and marketing strategies, to allow resource allocation, and improve efficiency in customer relationships. This chapter briefly summarizes the thesis that a Stable Data Mining Framework can be built using SSM and SPL. It introduces the challenges faced during the research, lists the things that would be done in the future, and provides the conclusions.

8.1 Challenges

Patterns and the KM with different scenarios and domains are validated and verified to make the data mining framework applicable across domains.

Any Mechanism pattern, which is the parent class for all algorithms, is provided. Hooks to add new algorithms, or activate any existing algorithm, or use the functionality of the tool are provided. These features allow the UDMS tool to support all the data mining algorithms.

Any Selection pattern is suggested, which provides the user with the list of algorithms according to the applicability and the option to select a choice. To make this selection, the user is provided with the description and the advantages of the algorithms. For example, algorithms such as Clustering [44 – 46, 69 -70] and Decision Trees [56,57] are useful in areas such as marketing and insurance companies, whereas Text Mining is useful in knowledge discovery [10,11] applications.

Implementations of all the data mining algorithms and data mining techniques have been provided. Hooks have been provided to attach these techniques and algorithms to the data mining framework. Depending on the application and the user's choice, hooks activate the algorithms and techniques so that no compromise is made in the performance of the tool to support all data mining algorithms and techniques.

Data Visualization [69,70] is suggested as remote KM to the data mining core KM. This way the data mining patterns can be represented using different visualization techniques in order to present data mining solutions.

8.2 Future Work

Modeling all the capabilities and goals

Modeling of other data mining capabilities such as Any Data Mining Application, Any Data Analysis, and their respective goals are left for future work. Also, modeling all the second level patterns for the Knowledge, Analysis, and all the capabilities is also left for future work.

Implementation of patterns

Implementation of patterns such as Knowledge, Analysis, Any Collection, Any Preparation, and Any Selection are left for future work.

Generation of architectures

There are a number of architectures that can be generated from the KM. The generation of architectures from this KM is left for future work.

Documenting specifications

Documenting specifications for all the goals and capabilities is also left for future work.

Documenting the patterns

Documentation for patterns such as Discovery and Any Data Mining are provided in Appendix A and Appendix B, respectively. The detailed documentation for the remaining patterns such as Knowledge, and Analysis is left for future work.

Documenting the framework

Documentation for the data mining framework and detailed description of the scope of the framework is left for future work.

Developing the prototype

The prototype for the tool called UDMS is left for future work. The prototype involves modeling and implementing 20 to 22 patterns. It also involves implementing hooks for the capabilities involved and implementing all the algorithms in a layered approach. For example, the algorithms, which are used in most applications and domains will be grouped in layer 1 and the following layers will contain the rest of the algorithms depending on their popularity and their usefulness.

Documenting the prototype

The documentation of the UDMS, its specifications, its scope, and its advantages is left for future work. Details about the layered approach for algorithms, the description, and advantages of all algorithms are left for future work.

8.3 Conclusions

The research involved in this thesis has given insight into SSM and how it can be used in data mining. This thesis suggests that with the applicability of SSM, a KM and Unified Data Mining Framework can be generated.

This data mining framework called UDMS, is stable because it is generated using Stable Analysis Patterns such as Discovery, Analysis, and Knowledge. The framework and the modeled patterns are applied across different domains and applications. UDMS can be customized using extension points called hooks so that the framework is applicable in any domain. To demonstrate the applicability and customization of UDMS, the framework has been applied to knowledge discovery and marketing applications. To demonstrate the stable nature of Discovery Analysis Pattern, applications such as discovery of a planet, and knowledge discovery were tested successfully. To demonstrate the stability of Any Data Mining Design Pattern, applications in the areas of Marketing, Prediction and Estimation of data were tested successfully. In addition, hooks used to customize the framework to be applicable in these applications are also provided.

Using UDMS, different architectures involving goals and capabilities can be generated. Table 8-1 summarizes the results generated from this thesis.

Table 8-1 Summary of Major Results

Major Results	
1	Modeled Stable Analysis Patterns such as Discovery, Knowledge, and Analysis.
2	Modeled Stable Design Patterns such as Any Data Collection, Any Data Preparation, Any Selection, and Any Data Mining.
3	Implemented patterns such as Discovery and Any Data Mining.
4	Extended the implemented patterns using hooks.
5	Generated applications such as knowledge discovery, discovery of a planet, credit card fraud detection application, and moviegoer's application using hooks.
6	Provided documentation for the implemented patterns (Discovery and Any Data Mining) and the modeled patterns.
7	Generated Data Mining KM using goals and its respective capabilities.
8	Provided description of the KM and its properties.
9	Provided description of the intersection and the remote KM.

References

[1] Westphal C, Blaxton T. Data Mining Solutions: Methods and Tools for Solving Real-World Problems. John Wiley and Sons; 1998.

[2] Fayad M. Accomplishing Software Stability. Communications of the ACM. 2001 Jan; 45(1).

[3] Fayad M. How to Deal with Software Stability. Communications of the ACM. 2002 Apr; 45(4).

[4] Goodman A. Introduction to Data Collection and Analysis [Online] [2003?] [cited 2005 Aug 7]; Available from: URL: http://www.deakin.edu.au/~agoodman/sci101/index.php

[5] Berry M, Linoff G. Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management. John Wiley and Sons; 1997.

[6] The Date Warehousing Information Center – A Definition of Data Warehousing [Online] [1995?] [cited 2005 Aug7]; Available from: URL: http://www.dwinfocenter.org/defined.html

[7] DataWarehousing.com – documenting data replication and data transformation sites on the Net [Online] [2004?] [cited 2005 Aug 21]; Available from: URL: http://www/datawarehousing.com/

[8] Webopedia: Online Computer Dictionary for Computer and Internet Terms and Definitions [Online]. Available from: URL: http://www.webopedia.com/

[9] Informatica's Technical Glossary [Online]. [cited 2005 Aug 25]; Available from: URL: http://www.informatica.com/solutions/resource_center/glossary/default.htm

[10] Wright P. Knowledge Discovery in Databases : Tools and Techniques [serial Online] 1998 [cited 2005 June 4]; Available from: URL: http://www.acm.org/crossroads/xrds5-2/kdd.html

[11] Attar Software. Data Mining with XpertRule Miner [Online]. [2002] [cited 2005 Aug 16] ; Available from: URL: http://www.intellicrafters.com/mineroverview.pdf

[12] Fayad, M, Schmidt D. Building Application Frameworks: Object-Oriented Foundations of Design. John Wiley and Sons; 1999.

[13] Froehlich G, Hoover J, Liu L, Sorenson P. Reusing application frameworks through hooks, in Object-Oriented Application Frameworks. Fayad M, Schmidt D, Johnson R, editors. John Wiley and Sons, 1999.

[14] Fayad M, Altman A. Introduction to Software Stability. Communications of the ACM 2001 Sept; 44(9):95-8.

[15] Fayad M, Hamza H. Stable Analysis Patterns: A True Problem Understanding with UML. Proceedings of the UML 2003 Full Day Workshop; 2003 Oct 20-24; San Francisco, USA.

[16] Salford Systems. CART [Online]. 2003 [cited 2005 July 7]; Available from: URL: http://www.salford-systems.com/cart.php

[17] Breiman L, Friedman J, Olshen R, Stone C. Classification and Regression Trees. Wadsworth Inc.; 1984.

[18] Khoshgoftaar TM, Allen EB. Modeling Software Quality with Classification Trees. Pham H, editor. World Scientific; 1999.

[19] MacQueen JB. Some Methods For Classification and Analysis of Multivariate Observations. In: L. M. LeCam and J. Neyman, editors. Proceedings of the Fifth Berkeley Symposium on Mathematical Statistic and Probability.University of California Press, Berkley; 1967.

[20] Data Mining: What is Data Mining? [Online]. [2003?] [cited 2005 July 7]; Available from: URL: http://www.anderson.ucla.edu/faculty/jason.frand/teacher/technologies/palace/datamining.htm

[21] Porter AA, Selby RW. Empirically Guided Software Development Using Metric-Based Classification Trees. IEEE Software. 1990 Mar; 7(2): 46-54.

[22] Porter AA, Selby RW. Evaluating Techniques for Generating Metric-based Classification Trees. J. Systems Software; 1990 Dec; 12(3): 209-218.

[23] Salford Systems. CART [Online]. 2003 [cited 2005 July 7]; Available from: URL: http://www.goldenmean.com.au/Documents/CARTtrifold.pdf/

[24] XpertRule Software Ltd. White Paper: Data Mining with Miner [Online]. [2004?] [cited 2005 July 7]; Available from: URL: http://www.attar.com/tutor/miner.htm

[25] Cross Industry Standard Process for Data Mining. Journal of Data Warehousing. The Data Warehousing Institute's Mission [serial online] 2000 Aug [cited 2005 July 7]; 5(4). Available from: URL: http://www.crisp-dm.org/News/86605.pdf

[26] Klösgen W, Zytkow JM. Knowledge Discovery in Databases Terminology. In: Fayyad UM, Piatetsky-Shapiro G, Smith P, Uthurusamy R, editors. Advances in Knowledge Discovery and Data Mining. AAAI Press/ The MIT Press; 1996.

[27] Fayyad UM, Piatestsky-Shapiro G, Smyth P, Uthurusamy R. Advances in Knowledge Discovery and Data Mining. AAAI Press/MIT Press; 1996.

[28] Fayyad UM, Piatetsky-Shapiro G, Smyth P, Uthurusamy R. The KDD process for extracting useful knowledge from volumes of data. Communications of the ACM. 1996 Nov; 39(11): 27-34.

[29] Fayyad UM, Uthurusamy R. Data mining and knowledge discovery in databases, Communications of the ACM. 1996 Nov; 39(11): 24-26.

[30] Quinlan JR. Induction of Decision Trees. Machine Learning. 1986; 1(1): 81-106.

[31] Cooper GF, Herskovitz E. A Bayesian Method for the Induction of Probabilistic Networks from Data. Machine Learning. 1992 ;(9): 309-347.

[32] White papers on Knowledge Management and Data Mining [Online]. [2004?] [cited 2005 Aug 10] ; Available from: URL: http://www.xpertrule.com/tutor/papers.htm

[33] XpertRule Software Ltd . [Online]. [2004?] [cited 2005 Aug 10] ; Available from: URL: http://www.xpertrule.com/tutor/mining.htm

[34] Data Mining and Discovery [Online]. [2005?] [cited 2005 Aug 01] ; Available from: URL: http://www.aaai.org/AITopics/html/mining.html

[35] Apptus Technologies AB. Apptus - You have the data, we have the technology [Online]. [2000?] [cited 2005 Aug 30]; Available from: URL: http://www.apptus.com/o.o.i.s/499

[36] Google [Online]. 1998 Sept 7. [cited 2005 Aug 25]; Available from: URL: http://www.google.com/

[37] New Millennium Resources Limited. Glossary of Terms. [Online] [2004?] [cited 2005 July 7]; Available from: URL: http://www.newmillennium/com.au/glossary.php

[38] Efron B. Estimating the Error Rate of a Prediction Rule: Improvement on Cross-Validation. Journal of the American Statistical Association. 1983 June; 78(382): 316-331.

[39] Jorgensen M. Experience With the Accuracy of Software Maintenance Task Effort Prediction Models. IEEE TSE. 1995 Aug21; (8): 674-681.

[40] Australian Government Department of Education and Training. [Online] [2005?] [cited 2005 Aug 8]; Available from: URL: http://www.dest.gov.au/default.htm

[41] Calc101.com Automatic Calculas, Linear Algebra, and Polynomials [Online]. [2002?] [cited 2005 Aug 10] ; Available from: URL: http://www.calc101.com/

[42] Thearling K. An Introduction to Data Mining: Discovering Hidden Value in your Data Warehouse [Online]. [1996?]; Available from: URL: http://www.thearling.com /text/dmwhit e/dmwhite.htm

[43] Pyle D. Data Preparation for Data Mining. Morgan Kaufmann Publishers, Inc.; 1999.

[44] Everitt BS,Landau S, Leese M. Cluster Analysis. Hodder Arnold Publication; 2001.

[45] Jain A, Dube, R. Algorithms for Clustering Data. Prentice-Hall; 1988.

[46] Milligan GW. An Examination of the Effect of Six Types of Error Perturbation of Fifteen Clustering Algorithms. Psychometrika 1980 Sept; 45(3): 325-342.

[47] Bishop CM. Neural Networks for Pattern Recognition. Oxford Press; 1995.

[48] Haykin S, Macmillan GW. Neural Networks A Comprehensive Foundation. Prentice Hall; 1998.

[49] Pattern Recognition Related Fields [Online] [2002?] [cited 2005 Aug 11] ; Available from: URL: http://www.ph.tn.tudelft.nl/PRInfo/fields.html

[50] Chester M. Neural Networks: A Tutorial. New Jersey: Prentice Hall; 1993.

[51] Michalewicz Z. Genetic Algorithms + Data Structures = Evolution Programs New York: Springer-Verlag; 1994.

[52] Goldberg DE. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Company; 1989.

[53] Holland J. Genetic Algorithms. Scientific American. 1992 July; 66-72.

[54] The MathWorks - Genetic Algorithm and Direct Search Toolbox - Solve optimization problems using genetic and direct search algorithms [Online] [1994?] [cited 2005 Aug 14]; Available from: URL: http://www.mathworks.com/products/gads/

[55] Mitchell M. An Introduction to Genetic Algorithms. MIT Press; 1996.

[56] Regression Tree - Introduction [Online] [1994?] [cited 2005 Aug 14]; Available from: URL: http://www.resample.com/xlminer/help/rtree/rtree_intro.htm

[57] Quinlan JR. C4.5: Programs for Machine Learning. Morgan-Kaufmann Publishers Inc.; 1993.

[58] Chambers JM, Hastie TJ. Statistical Models in S. Wadsworth International Group; 1992.

[59] Jelinek F. Statistical Methods for Speech Recognition. MIT Press; 1992.

[60] Kaski S. Methods for Exploratory Data Analysis [Online]. [1997?] [cited 2005 Aug 15] ; Available from: URL: http://www.cis.hut.fi/~sami/thesis/node7.html

[61] University College London. Frozen Sea Discovered Near Martian Equator From 3D Images Of Mars Express [Online]. 2005 Mar 2 [cited 2005 Aug 30]; Available from: URL: http://www.sciencedaily.com/releases/2005/02/050224112321.htm

[62] Center for Global Food Issues. Mad Cow Disease – Bovine Spongiform Encephalopathy: BSE Facts [Online]. 2004 July 24 [cited 2005 July 5]; Available from: URL: http://www.mad-cowfacts.com/about.htm

[63] Brown A, IBM. An introduction to Model Driven Architecture Part I: MDA [Online]. [2004] [cited 2005 Aug 15] ; Available from: URL: http://www-128.ibm.com/developerworks/rational/library/3100.html

[64] Brown A. An Introduction to Model Driven Architecture. Part 1: MDA and today's systems [serial online] 2004 Feb 17 [cited 2005 June 4]; Available from: URL: http://www-128.ibm.com/developerworks/rational/library/3100.html

[65] Scianta Intelligence. Data Mining [Online]. 2005 [cited 2005 Aug 30]; Available from: URL: http://scianta.com/technology/datamining.htm

[66] Java Technology [Online]. [1994] [cited 2003 Jan 15] ; Available from: URL: http://www.sun.com/java/

[67] Apache Tomcat [Online]. [1999] [cited 2003 Jan 18] ; Available from: URL: http://www.tomcat.apache.org/

[68] VitaK Inc VitaK Discovery [Online]. [2001?] [cited 2005 Aug 18] ; Available from: URL: http://www.vitak.com/index.php?id=24

[69] Tufte E, The Visual Display of Quantitative Information. Graphics Press; 1983.

[70] Wainer H. Visual Revelations. Copernicus; 1997.

APPENDIX A Stable Analysis Pattern - Discovery Pattern

A.1 Pattern Documentation

Pattern Name - Discovery

The term 'Discovery' represents an act of finding something. Discovery may be a productive insight causing a breakthrough in some domain or a compulsory revelation of facts. Thus, Discovery is an enduring concept. This concept can be represented by a Discovery pattern.

Known As

Discovery is also known as Innovation and Invention [1], but it has different meanings in different scenarios. For example, Discovery in a Data Mining scenario is knowledge discovery, but discovery of the number '0' by Aryabhatta is an invention.

Context

This pattern can be used in any application where a discovery concept applies. It can be used for discovery of facts, patterns, or discovery of anything in a multitude of domains.

Problem

Discovery is an enduring concept whose application ranges from discovery of universe, to discovery of a mathematical formula, to discovery of patterns in data. Currently, several software applications utilize the discovery concept in a variety of tools to discover software bugs, discover and diagnose diseases, discover failures in controlled systems, and discover evidences in research. The discovery concept utilized in these software applications shares little similarity in code structure. Therefore, building a new

system from an existing one requires rewriting the code. This makes it difficult to effectively reuse the code from previous systems.

Challenges

Discovery pattern is used to model the components surrounding the discovery concept, which can perform discovery in a variety of domains or field. However, the very definition of discovery needs to be constrained for this pattern. The claim to discover is different from the act of invention, which needs to be addressed during modeling of this pattern.

Discovery is a generic concept – it can mean any breakthrough by serendipity or a product of persistent research and innovation. These flavors of discovery need to be handled by the stable pattern.

Discovery also has various forms in different domains. For example, discovery of water on Mars is conducted by Observation and Research and not by Experimentations, whereas discovery of a pattern is conducted using Experiments, Observations, and Examinations. All these different aspects need to be addressed when building a model.

Discovery has multiple types. Some discovery types go hand in hand with other discovery types to discover artifacts. For the most effective reuse of the discovery pattern, it must be coupled with a variety of hooks in the "BO" type patterns and classes. These hooks serve as tools to extend the BO class through associations, inheritances, and aggregation into a specific domain application.

Constraints

Depending on the type of discovery, there could be one or more discovery mechanisms that need to be activated. For example, discovery of a medicine requires Observation, Experimentation, and Research.

An individual or a group of individuals conducts the discovery process. For example, scientists collaborate with doctors, analysts, and researchers to achieve a solution. The discovery mechanism leads to one or more evidences. AnyEvidence indicates a discovery or discoveries. The discovery process should qualify the discovery or discoveries made.

Solution

The solution to the above problem is demonstrated in the form of a model, followed by a discussion on the participants. This solution provides a reusable software application framework pattern that can be utilized across a variety of software applications requiring the discovery concept.

Pattern Structure

Discovery exists in different fashions across many domains, such as discovery of a planet, or a star, or discovery of a new medicine. It can be categorized as a geological discovery, medical discovery, and knowledge discovery. Depending on the type of discovery, there could be either one or many discovery mechanisms. These mechanisms could be Examination, Experimentation, Observation, or Research. An individual, or a group of individuals, or a company conducts the discovery process. Every discovery requires the evidence to prove itself. The discovery mechanism leads to the evidence,

which finally proves the discovery or discoveries. This discovery should meet the criteria for which it was discovered and needs to be qualified to be a valid discovery. Figure A-1 puts together the relationship between the above concepts.

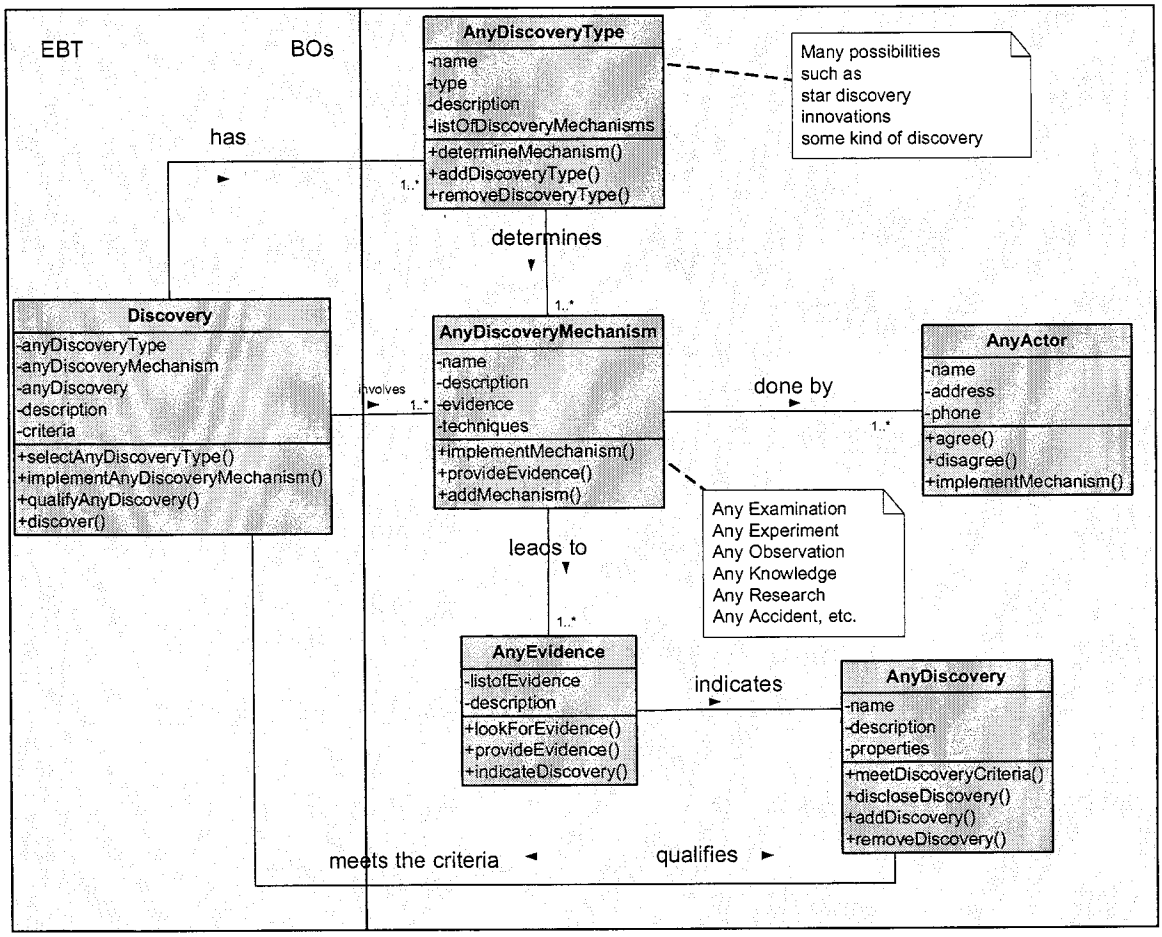


Figure A-1 Discovery Stable Analysis Pattern

Participants

The participants in the Discovery pattern are classified in two categories, class and patterns. The pattern classification has a second level of abstraction containing actual class representation.

Patterns

Any Discovery Type – Represents the different types of discoveries in different application areas or domains.

Any Discovery Mechanism - Represents the BO, which deals with different kinds of discovery mechanisms.

Any Discovery - Represents the BO, which represents the desired discovery.

Any Actor - Represents a person, or a group of people that interacts, or scientist group responsible for the discovery process.

Any Evidence - Represents the proof of the discovery.

Class

Discovery - Describes the discovery process.

CRC – Cards

CRC cards are filling cards on which the class, its responsibilities and other collaborators are noted. A class is responsible for the knowledge its objects should have (attributes) and for the operations its objects must carry out to fulfill their tasks and reach their goals. Collaborators are other classes for which relationships must exist to enable the class to fulfill its tasks.

CRC - Card 1: Discovery (Finding)		
Responsibility	**Collaboration**	
Discovering	**Client**	**Server**
	AnyDiscoveryType AnyDiscoveryMechanism AnyDiscovery	implementAnyDiscoveryMechanism() qualifyAnyDiscovery() discover() selectAnyDiscoveryType()

CRC - Card 2: Any Discovery Mechanism (Implementation)		
Responsibility	Collaboration	
To implement	Client	Server
	Discovery Any Discovery Type Any Actor Any Evidence	implementMechanism() provideEvidence() addMechanism()

CRC - Card 3: AnyActor (Perform)		
Responsibility	**Collaboration**	
To perform	**Client**	**Server**
	Any Discovery Mechanism	agree() disagree() implementMechanism()

CRC - Card 4: Any Discovery Type (Classify)		
Responsibility	**Collaboration**	
To classify the types of discoveries	**Client**	**Server**
	Discovery AnyDiscoveryMechanism	addDiscoveryType() determineMechanism() removeDiscoveryType()

CRC - Card 5: Any Evidence (Proof)		
Responsibility	**Collaboration**	
To provide proof	**Client**	**Server**
	AnyDiscovery AnyDiscoveryMechanism	provideEvidence() lookForEvidence() indicateDiscovery()

CRC - Card 6: Any Discovery (Element)		
Responsibility	**Collaboration**	
Store information about itself	**Client**	**Server**
	Discovery AnyEvidence	meetDiscoveryCriteria(), discloseDiscovery(), addDiscovery(),

Consequences

This pattern supports the motivation behind its modeling. It depicts a generic pattern, which can be utilized in applications across diverse domains.

A.2 Applicability: Case Study 1 - Discovery of Vitamin K

This case study demonstrates the applicability of the Discovery pattern in the discovery of Vitamin K. Vitamin K discovery is a part of Any Discovery Type. The different discovery mechanisms determined are Experiments, Investigation and Research. The scientists involved in this discovery process are responsible for initiating the discovery process, conducting the Research and Experiments. Scientist inherits information from super class Any Actor. The results of these mechanisms act as evidence and Vitamin K is a part of Any Discovery.

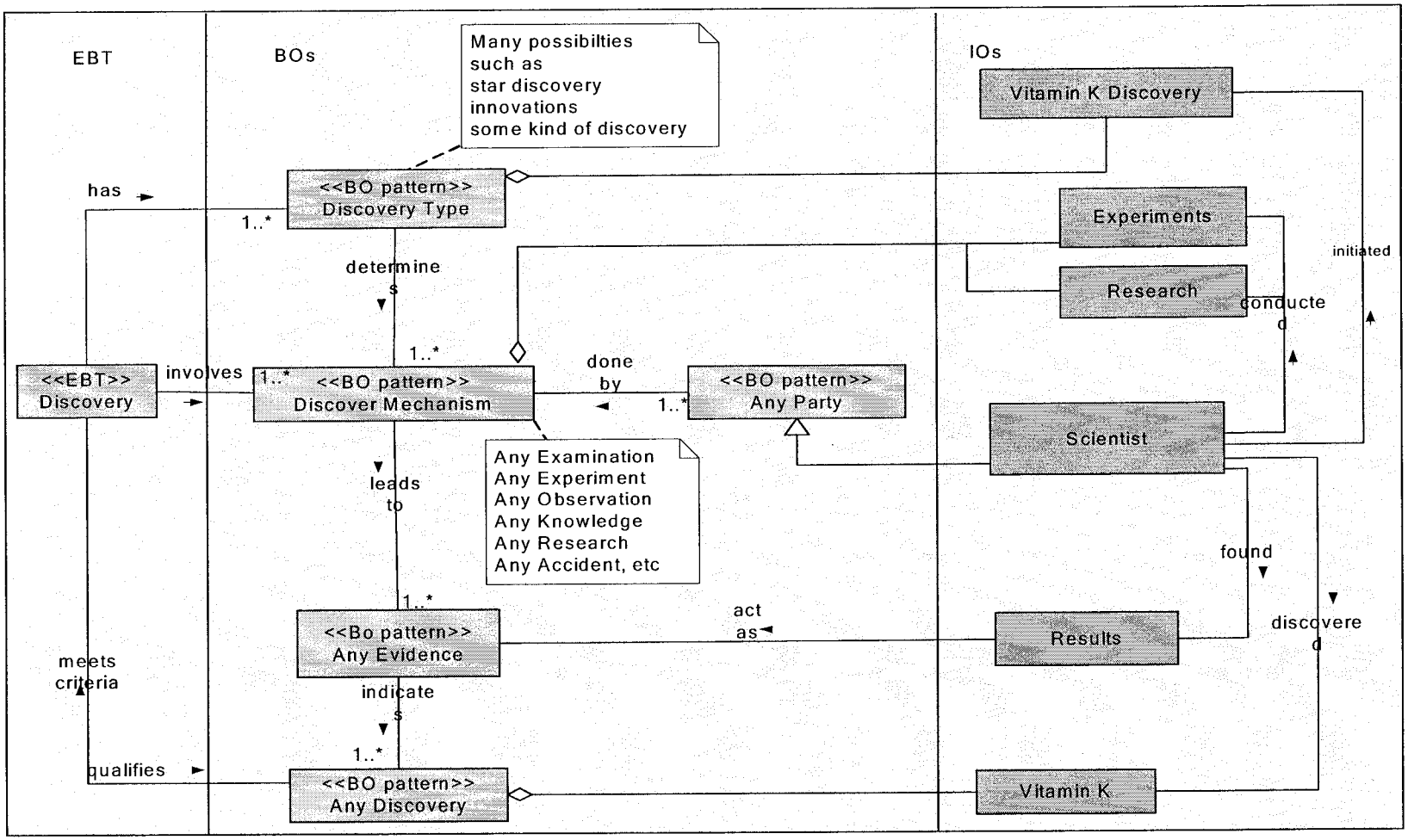


Figure A-2 Class Diagram for Case Study 1

A.2.1 Use Case Description

Boxes that follow use the use case template to describe the use case for discovery of Vitamin K.

Use Case Id:	**Use Case Title:**
1.1	Discovering Vitamin K.

Actors	**Roles**
Any Discovery Type	Categorize
Any Discovery Mechanism	Technique
Any Evidence	Proof
Any Discovery	Discovery
Any Party	Doctor

Classes	**Attributes**	**Operations**
Discovery	description listOfFindings	implementAnyDiscoveryMechanism() qualifyAnyDiscovery() discover()
Any Discovery Type	name type description	addDiscoveryType() determineMechanism() removeDiscoveryType()
Any Discovery	name properties description	meetDiscoveryCriteria() discloseDiscovery() addDiscovery()
Any Discovery Mechanism	name description evidence techniques	implementMechanism() provideEvidence() addMechanism()

Any Evidence	listOfEvidence description	provideEvidence() lookForEvidence() indicateDiscovery()
Any Actor	name address phone	agree() disagree() implementMechanism()
Vitamin K Discovery	description characteristics	intitiateDiscovery() determineMechanism()
Vitamin K	name qualities	listProperties() listAilments() listAudience() listMedications()
Experiments	data statistics results attributes	extractInfo() provideProof() recordResults()
Investigation	data statistics results attributes	extractInfo() provideProof() recordResults()
Research	data statistics results	extractInfo() provideProof() recordResults()
Scientist	skills publications researchInfo	listResearchArea() initiateDiscovery() initiateResearch()
Results	description	provideResults() provideDescription()

EBTs	BOs	IOs
Discovery	Any Discovery	Experiments
	Any Discovery Mechanism	Results
	Any Party	Scientist
	Any Evidence	Vitamin K
	Any Discovery Type	Investigation
		Research
		Vitamin K discovery

Use Case Description	1. Scientist (Any Party) initiates Vitamin K discovery (Any Discovery Type). 2. Vitamin K discovery (Any Discovery Type) determines the mechanisms, which are Experiments, Investigation, and Research (Any Discovery Mechanism). 3. Scientist (Any Party) conducted the Experiments, Investigations, and Research (Any Discovery Mechanism). 4. Any Discovery Mechanism leads to Any Evidence, which are the Results of the mechanisms. 5. These Results indicate the discovery, which is Vitamin K.

A.2.2 Behavior Diagram

Figure A-3 represents the sequence diagram for the use case provided for the above application.

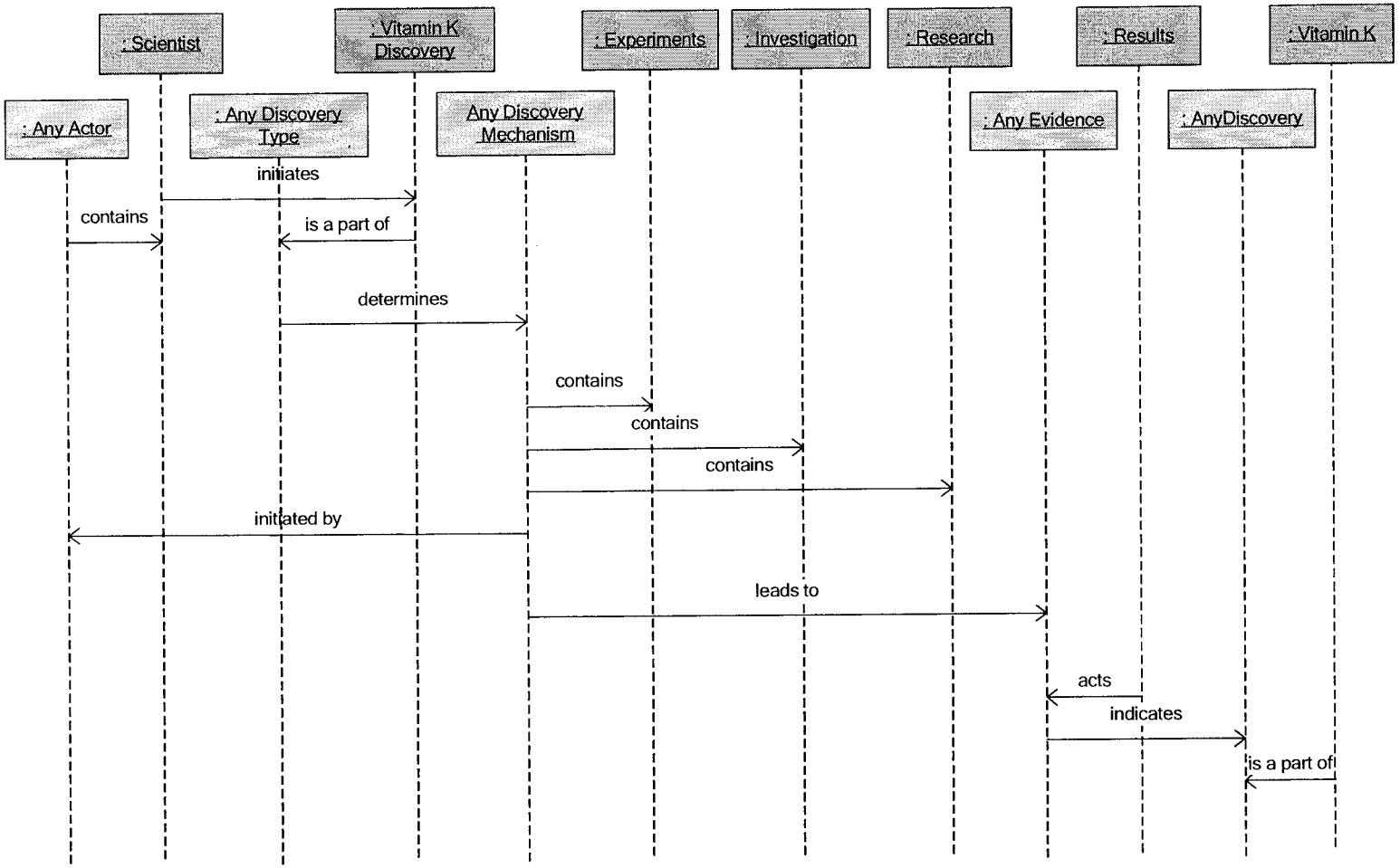


Figure A-3 Sequence Diagram for Discover Vitamin K Use Case.

A.3 Applicability: Case Study 2 - Research Application

This case study demonstrates the applicability of the Discovery pattern in a research application used by scientists to track information gathering in a particular field of science.

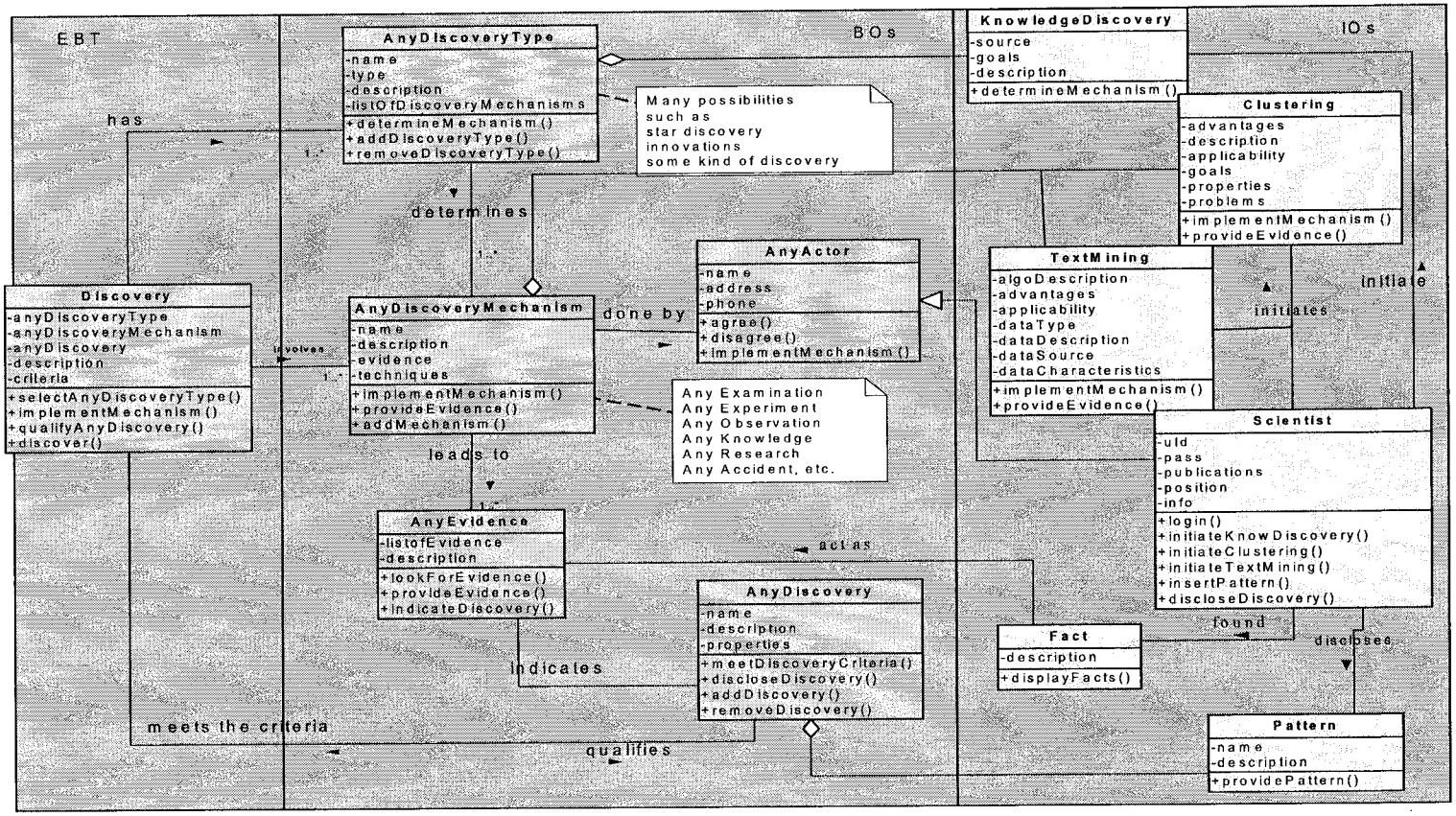


Figure A-4 Knowledge Discovery Application

A.3.1 Use Case Description

Boxes that follow use the use case template to describe the use case for knowledge discovery application.

Use Case Id:	Use Case Title:
2.1	Knowledge Discovery

Actors	Roles
Any Discovery Type	Categorize
Any Discovery Mechanism	Technique
Any Evidence	Proof
Any Discovery	Discovery
Any Party	Doctor

Classes	Attributes	Operations
Discovery	description listOfFindings	implementAnyDiscoveryMechanism() qualifyAnyDiscovery() discover()
Any Discovery Type	name type description	addDiscoveryType() determineMechanism() removeDiscoveryType()
Any Discovery	name properties description	meetDiscoveryCriteria() discloseDiscovery() addDiscovery()
Any Discovery Mechanism	name, description techniques	implementMechanism() provideEvidence() addMechanism()
Any Evidence	listOfEvidence description	provideEvidence() lookForEvidence() indicateDiscovery()

Any Actor	name address phone	agree() disagree() implementMechanism()
Knowledge Discovery	source rules	determineMechanism
Pattern	name description	providePattern()
Clustering	advantages description applicability result	implementMechanism() provideEvidence()
Text Mining	algoDescription advantages applicability dataType	implementMechanism() provideEvidence()
Scientist	uId pass publications position info	login() initiateKnowDiscovery() initiateClustering() initiateTextMining() insertPattern() discloseDiscovery()
Fact	description	displayFacts()

EBTs	**BOs**	**IOs**
Discovery	Any Discovery Any Discovery Mechanism Any Party Any Evidence Any Discovery Type.	Clustering Text Mining Scientist Pattern Knowledge Discovery Fact

Use Case Description	1. Scientist (Any Party) initiates Knowledge Discovery (Any Discovery Type). 2. Knowledge Discovery (Any Discovery Type) determines the use of TextMining and Clustering (Any Discovery Mechanism) as the best mechanism to be used for Knowledge Discovery. 3. Scientist (Any Party) conducted the Knowledge Discovery using Clustering and TextMining (Any Discovery Mechanism). 4. Clustering and TextMining (Any Discovery Mechanism) lead to Facts (Any Evidence). 5. The Scientists record the discovery.

A.3.2 Behavior Diagram

Figure A-5 represents the sequence diagram for the use case provided for the above application.

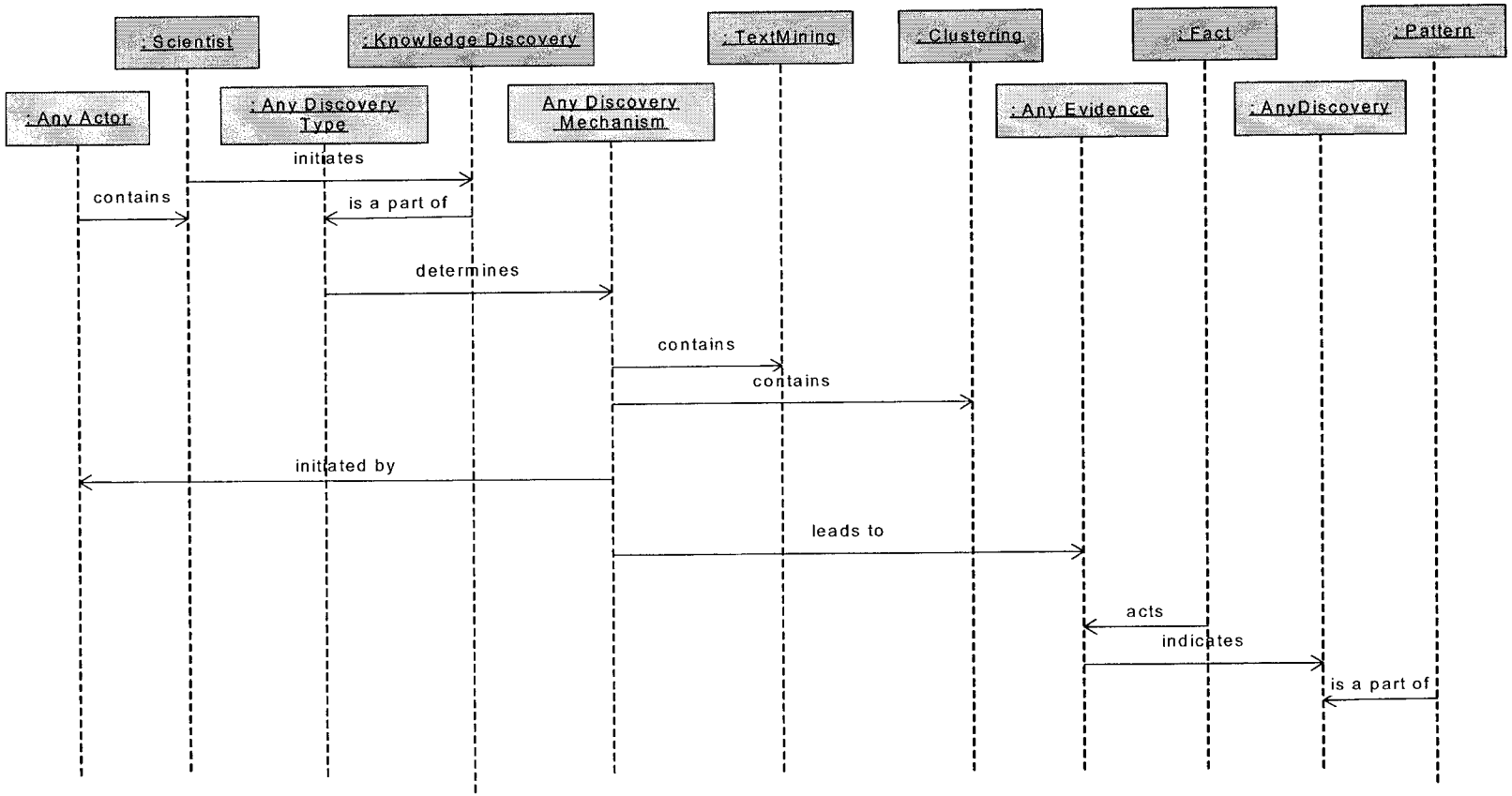


Figure A-5 Knowledge Discovery Sequence Diagram

A.4 Applicability: Case Study 3 - Planetary Research

This case study demonstrates the applicability of the Discovery pattern being used in a planetary research application.

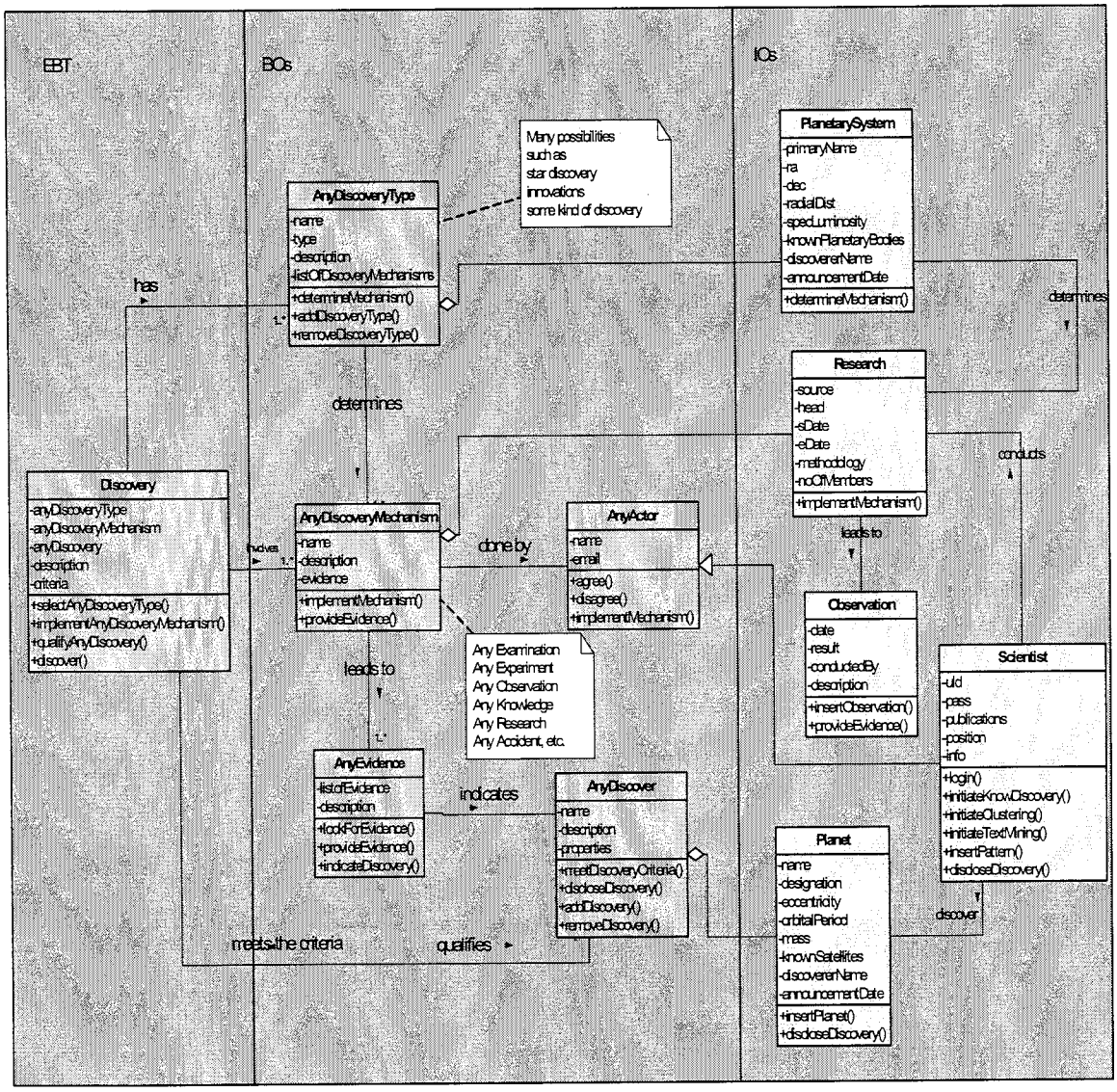


Figure A-6 Planetary System Application

A.4.1 Use Case Description

Boxes that follow use the use case template to describe the use case for discovery of a planet.

Use Case Id:	Use Case Title:
3.1	Discovery of a Planet

Actors	Roles
Any Discovery Type	Categorize
Any Discovery Mechanism	Technique
Any Evidence	Proof
Any Discovery	Discovery
Any Party	Doctor

Classes	Attributes	Operations
Discovery	description listOfFindings	implementAnyDiscoveryMechanism() qualifyAnyDiscovery() discover() selectAnyDiscoveryType()
Any Discovery Type	name type description	addDiscoveryType() determineMechanism() removeDiscoveryType()
Any Discovery	name properties description	meetDiscoveryCriteria() discloseDiscovery() addDiscovery()
Any Discovery Mechanism	name description techniques	implementMechanism() provideEvidence() addMechanism()
Any Evidence	listOfEvidence description	provideEvidence() lookForEvidence() indicateDiscovery()

Any Actor	name address phone	agree() disagree() implementMechanism()
Planetary System	primaryName ra dec radialDist specLuminosity knownPlanetaryBodies discovererName announcemanetDate	determineMechanism()
Planet	name designation eccentricity orbitalPeriod mass knownSatellites discovererName announcementDate	insertPlanet() discloseDiscovery()
Research	source head sDate eDate	implementMechanism()
Observation	date result conductedBy description	insertObservation() provideEvidence()
Scientist	uId pass publications position	login() initiatePlanetarySystem() insertPlanet() discloseDiscovery()

EBTs	BOs	IOs
Discovery	Any Discovery	Research
	Any Discovery Mechanism	Observation
	Any Party	Scientist
	Any Evidence	Planet
	Any Discovery Type	Planetary System

Use Case Description	1. Scientist (Any Party) initiates Planetary Discovery (Any Discovery Type). 2. Planetary Discovery (Any Discovery Type) determines the use of Research and Observation (Any Discovery Mechanism) as the best mechanism to be used for Planetary Discovery. 3. Scientist (Any Party) conducted the Planetary Discovery using Research and Observation (Any Discovery Mechanism). 4. Research and Observation (Any Discovery Mechanism) lead to Planet discovery (Any Discovery).

A.4.2 Behavior Diagram

Figure A-7 represents the sequence diagram for the use case provided for the above application.

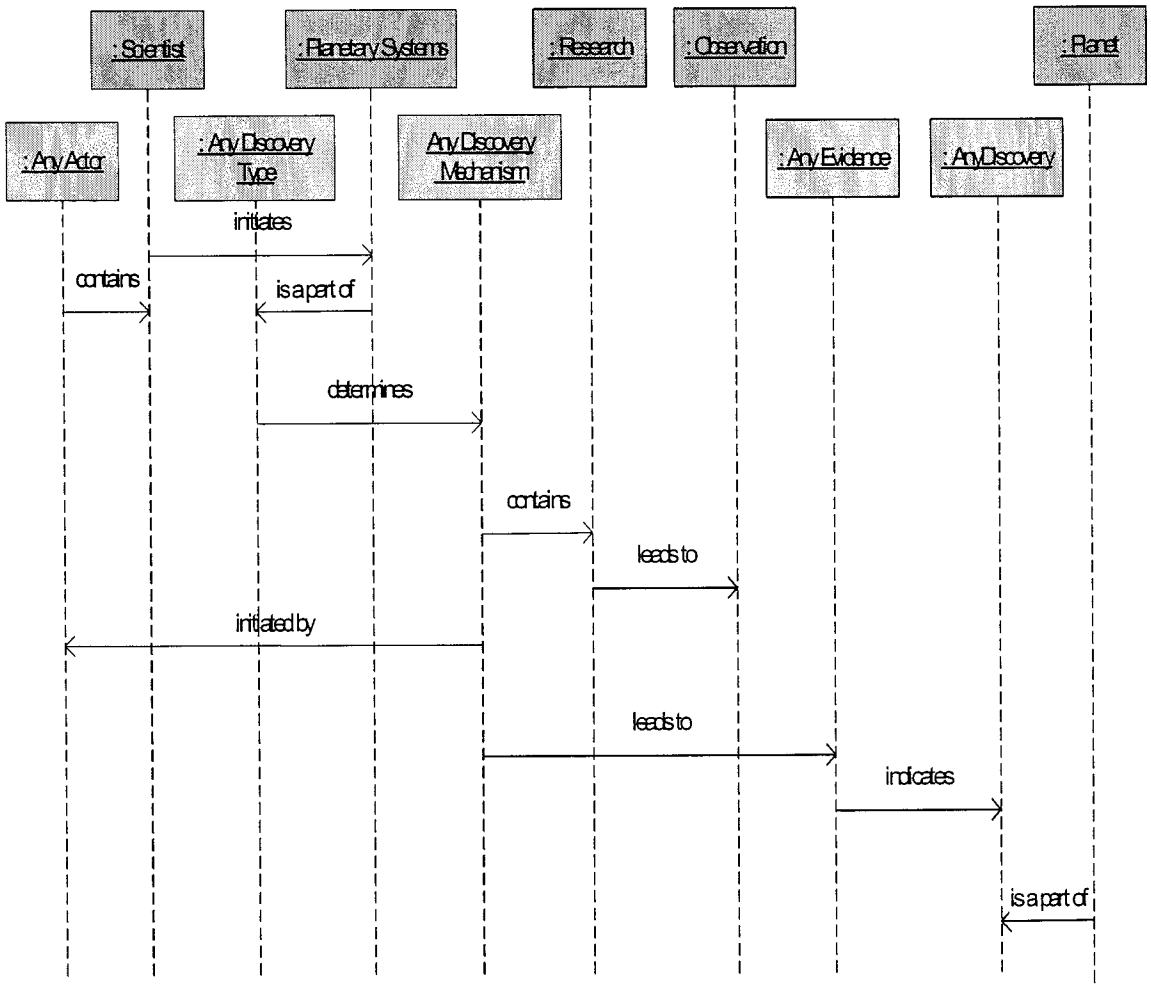


Figure A-7 Sequence Diagram for Discovery of a Planet

APPENDIX B Stable Design Pattern - Any Data Mining Pattern

B.1 Pattern Documentation

Pattern Name – Any Data Mining

Data mining is an information extraction activity whose goal is to analyze data and discover hidden facts contained in databases [2]. As a result, the pattern is named Any Data Mining pattern.

Context

This pattern can be used in any application and in any domain where data mining is required. This is a generic pattern, which is modeled considering various scenarios and different applications. As a result, a Stable Design Pattern is generated, which can be used and reused in different applications.

Problem

Data mining has different data mining techniques, such as Data Farming, Data Preparation, Data Collection, and Data Analysis. Data mining also has different processes such as Data Exploration, Discovery of Data Patterns, Data Collection, and Data Preparation. In addition to different data mining techniques and data mining processes, Data mining implements many algorithms such as Clustering [37 - 39], Neural Networks [40 - 43], Genetic Algorithms [44 - 48], Decision Trees [49,50], Classification Trees [30,49,53 -56], and other statistical methods [51,52]. The available data mining applications that are being utilized in the market may share little similarity in code structure. Depending on the application domain, the data mining applications vary and they provide different algorithms and techniques.

The main problem is to build a model that captures all the algorithms and data mining techniques available on the market. Another problem is to provide the model, which allows the addition or deletion of new or old algorithms and techniques.

Challenges

One challenge faced while modeling this pattern was to capture the different data mining techniques available on the market. Another challenge faced, was to represent the different state-of-the-art algorithms and provide the feature to add them.

Also, it was challenging to represent the results of data mining, as sometimes the result of using a data mining application was generation of a pattern, or generation of relationships, or associations. Data mining application can also be used for Prediction or Estimation of data. To represent the result was a challenging activity.

Another challenge was to add any new algorithms to the existing data mining software. Also, it was challenging to provide multiple algorithms and compare data to check similarity and differences.

Constraints

Any Data Mining leads to Analysis of data. Depending on the data mining application, the analysis of data could be different. For example, knowledge discovery data mining application can analyze data using Exploratory Data Analysis or Market Based Analysis. Analysis analyzes data (Any Data). This data can be unstructured data (raw data) or structured data such as XML file. Any Collection consists of data (Any Data), which can be represented in different forms. Collection can collect data such as stamps, books, or coins.

Any Discovery is discovered from one to many Collections (Any Collection) using different discovery mechanisms. For example, consider the universe, which is a collection of all planets and stars, and using different discovery techniques such as Observation, a new planet is discovered. Any Data Mining implements one to many mechanisms (Any Mechanisms), which can be algorithms such as Clustering, Neural Networks, or data mining techniques such as Unsupervised Learning or Supervised Learning. One to many Any Actor requests Any Data Mining. An actor can be a system or an individual, which requests or initiates the data mining process.

Solution

The solution to above stated problems is demonstrated in Any Data Mining pattern. This solution provides a reusable design pattern that can be used in different software applications requiring data mining.

Pattern Structure

Any Data Mining is a BO, which leads to Analysis of data, so we have the goal of data mining as Analysis and BO Any Data. For example, the different types of analysis are Exploratory Data Analysis, Market Based Analysis and Memory Based Analysis. Any Data is a part of Collection. For example, it could be from a database or from a data mart, so we have BO Any Collection. Any Discovery (a pattern or any relationship) is discovered from the collection. Any Data Mining implements different mechanisms such as Clustering, and Neural Networks. As a result, we have the BO Any Mechanism. Also, to represent the system or organization, which requests the data mining, we have

BO Any Actor. Figure B-1 represents the UML model for Any Data Mining Stable Design Pattern.

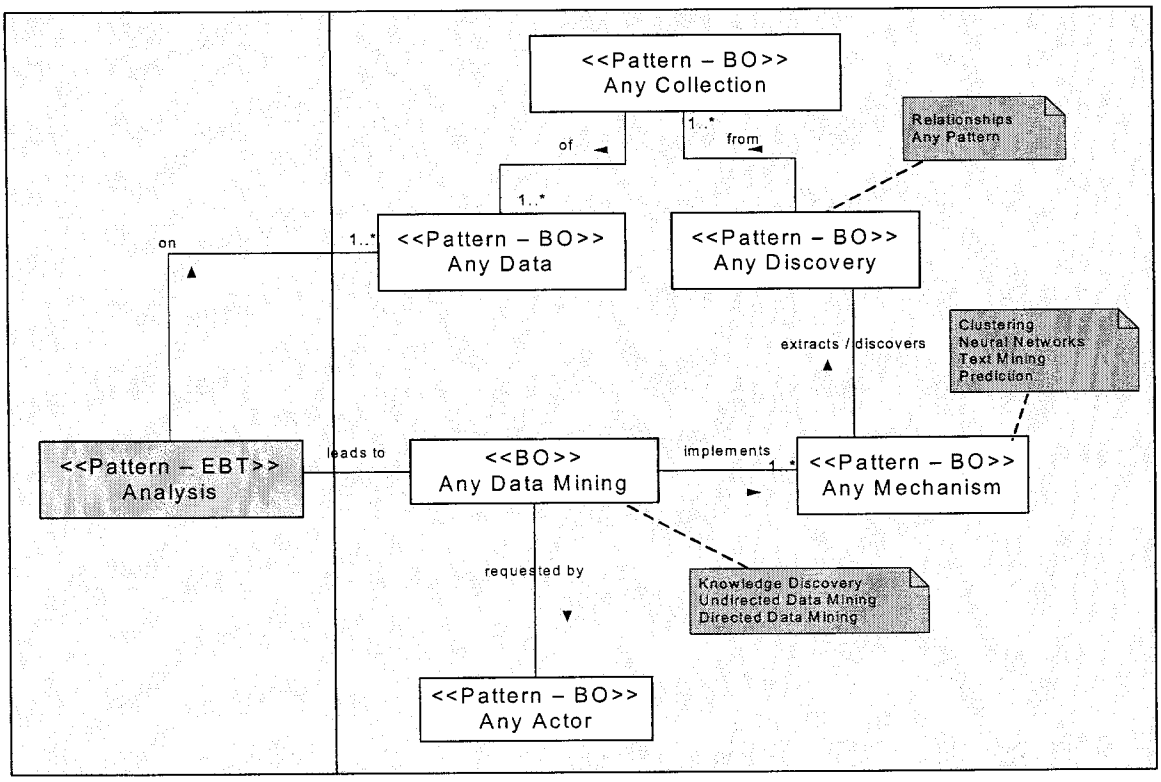


Figure B-1 Any Data Mining Pattern (taken from Dr. Fayad's Pattern archive)

Participants

The participants in the Any Data Mining pattern are classified in two categories, patterns and class.

Patterns

Analysis - Represents the EBT, which is the goal of data mining.

Any Actor - Represents any person, or a group of people, or any system.

Any Data - Represents the data, which is analyzed and collected.

Any Collection - Represents the act of gathering something together.

Any Discovery - Represents the results of data mining such as relationships, associations, or any patterns.

Any Mechanism - Represents the different data mining algorithms, such as Clustering and Neural Networks. This pattern also represents the data mining techniques and different data mining processes.

Class

Any Data Mining – Represents an information extraction activity whose goal is to analyze data and discover hidden facts contained in databases [2].

CRC – Cards

CRC cards are filling cards on which the class, its responsibilities and other collaborators are noted. A class is responsible for the knowledge its objects should have (attributes) and for the operations its objects must carry out to fulfill their tasks and reach their goals. Collaborators are other classes for which relationships must exist to enable the class to fulfill its tasks.

CRC - Card 1: Analysis (Analysis)		
Responsibility	Collaboration	
Analyze data	Client	Server
	Any Data Any Data Mining	analyzeData() initiateDM()

CRC - Card 2: Any Collection (Collector)		
Responsibility	**Collaboration**	
To collect	Client	Server
	Any Data Any Discovery	grow() shrink() organize() collectData()

CRC - Card 3: Any Actor (Perform)		
Responsibility	**Collaboration**	
To perform	**Client**	**Server**
	Any Data Mining	access() request()

CRC - Card 4: Any Data (Information)		
Responsibility	**Collaboration**	
To store information about itself	**Client**	**Server**
	Analysis Any Collection	storeInformation()

CRC - Card 5: Any Discovery (Result)		
Responsibility	**Collaboration**	
To represent Discovery	**Client**	**Server**
	Any Collection Any Mechanism	representDiscovery()

CRC - Card 6: Any Data Mining (DataMining)		
Responsibility	**Collaboration**	
To mine data	**Client**	**Server**
	Analysis Any Mechanism Any Actor	analyzeData() implementMechanism()

CRC - Card 7: Any Mechanism (Implementation)		
Responsibility	**Collaboration**	
To implement Mechanism	**Client**	**Server**
	Any data Mining Any Discovery	implementMechanism()

Consequences

This pattern supports the motivation behind its modeling. It depicts a generic pattern, which can be utilized in applications across diverse domains.

B.2 Applicability: Case Study 1 - Moviegoer's Application

Moviegoer's application uses Market Based Analysis and Exploratory Data Analysis to analyze data and to predict and estimate results. Market Based Analysis implements Prediction and Exploratory Data Analysis implements Estimation algorithm. These algorithms work on raw data and generate results.

B.2.1 Use Case Description

Boxes that follow use the use case template to describe the use case for mining data.

Use Case Id	Use Case Title
1.1	Mine Data

Actors	Roles
Any Collection	Accumulation
Analysis	Analysis

Any Mechanism	Implementation
Any Discovery	Discovery
Any Actor	User
Any Data	Information
Any Data Mining	Data Mining

Classes	**Attributes**	**Operations**
Analysis	noOfElements position size	analyze() initiateDM()
Any Actor	name address phone	access() request()
Any Collection	owner type description	grow() shrink() organize()
Any Data	characteristics type	storeInformation()
Any Data Mining	type description	analyzeData() implementMechanism()
Any Mechanism	name characteristics description	implementMechanism()
Any Discovery	name source description	representDiscovery()
User	name email	initiateApplication()

	phone	
DatabaseBean	connection	getConnection() queryDatabase() provideResults
Estimation	description definition	estimate() estimateSurvey() estimatePeopleperMovie()
Prediction	description applicability	predictMaleFrequency() predictMalesPerMovie() predictFemalesPerMovie() predictFemalesPerMovie()
Results	description	provideResults()
Exploratory Data Analysis	description focus philosophy	startEstimation()
Market Based Analysis	description advantages applicability problems	startPrediction()

EBTs	**BOs**	**IOs**
Analysis	Any Actor Any Data Mining Any Data Any Collection Any Discovery Any Mechanism	DatabaseBean Results Prediction Estimation Exploratory Data Analysis Market Based Analysis User

Use Case Description	1. The user initiates the Exploratory Data Analysis (EDA) and Market Based Analysis (MBA). EDA and MBA inherit default behavior from Any Data Mining. 2. Market Based Analysis starts Prediction and Exploratory Data Analysis starts Estimation. Prediction and Estimation are a part of Any Mechanism. 3. The algorithms provide Results. Result is a part of Any Discovery. 4. These results are obtained from the DatabaseBean. DatabaseBean represents Any Collection.

Class Diagram:

Figure B-2 represents the class diagram for the application scenario explained above.

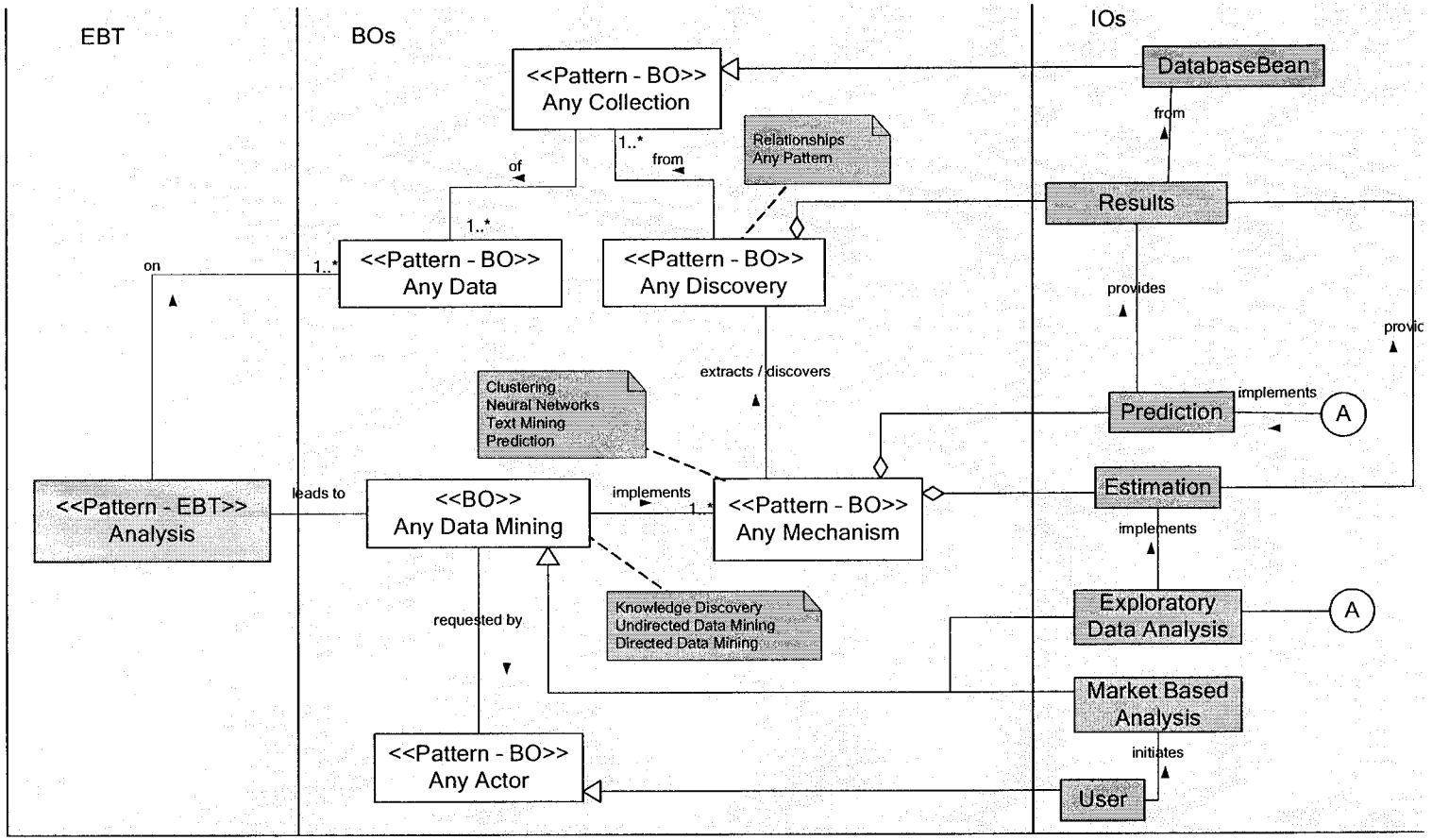


Figure B-2 Any Data Mining Pattern in Moviegoer's Application

B.2.2 Behavior Diagram

Figure B-3 represents the sequence diagram for the use case provided for application 1.

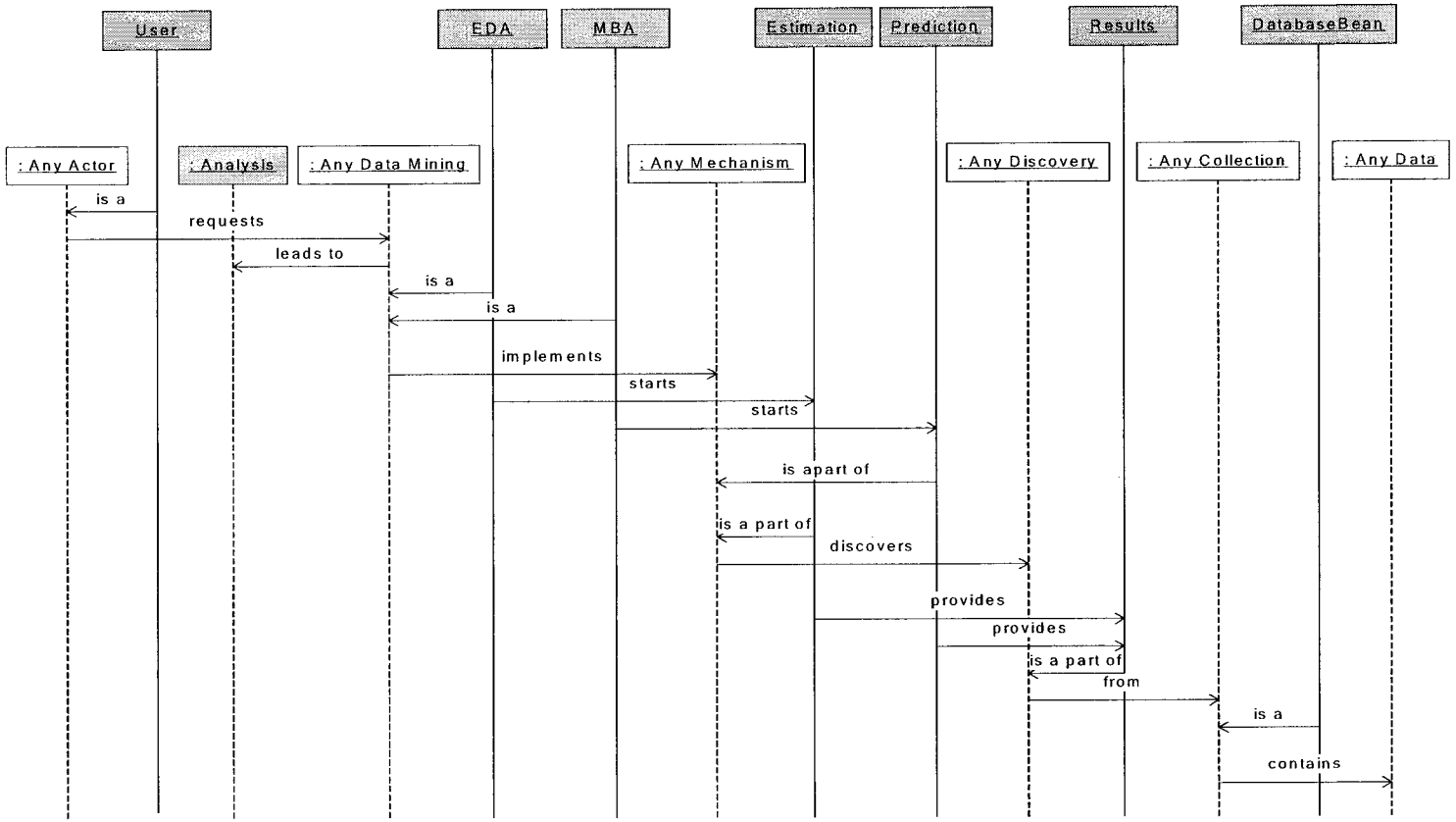


Figure B-3 Sequence Diagram for Application 1

B.3 Applicability: Case Study 2 - Credit Card Fraud Detection Application

This application illustrates Any Data Mining pattern's use and applicability in credit card fraud detection.

B.3.1 Use Case Description

Boxes below use the use case template to describe the use case for credit card fraud detection application.

Use Case Id	Use Case Title
2.1	Detect Fraud

Actors	Roles
Any Collection	Accumulation
Analysis	Analysis
Any Mechanism	Implementation
Any Discovery	Discovery
Any Actor	User
Any Data	Information
Any Data Mining	Data Mining

Classes	Attributes	Operations
Analysis	noOfElements size	analyze() initiateDM()
Any Actor	name address phone email	access() request()

Any Collection	owner type description	grow() shrink() organize()
Any Data	characteristics description type	storeInformation()
Any Data Mining	type description	analyzeData() implementMechanism()
Any Mechanism	name characteristics description	implementMechanism()
Any Discovery	name source description characteristics	representDiscovery()
User	name email phone#	initiateApplication()
DatabaseBean	connection	getConnection() queryDatabase() provideResults
Clustering	advantage applicability goals	implementMechanism() selectCustomerTransaction() provideEvidence()
Relationship	description source	provideResults()
UnSupervised Learning	definition description	initiateClustering() initiateDetailedClustering()

	applicability advantages	
Undirected DataMining	description properties advantages	startlearning()

EBTs	**BOs**	**IOs**
Analysis	Any Actor	DatabaseBean
	Any Data Mining	Relationship
	Any Data	Clustering
	Any Collection	UnSupervisedLearning
	Any Discovery	UnDirected DataMining
	Any Mechanism	User

Use Case Description	1. The user initiates the Undirected Data Mining (UDM). UDM inherits default behavior from Any Data Mining. 2. UDM starts UnSupervised Learning (USL). USL starts Clustering. Clustering is a part of Any Mechanism. 3. The algorithm provides Relationship. Relationship is a part of Any Discovery. 4. These relationships are obtained from the DatabaseBean. DatabaseBean represents Any Collection.

Class Diagram

Figure B-4 represents the class diagram for application 2. It represents the applicability of Any Data Mining pattern in credit card fraud detection application.

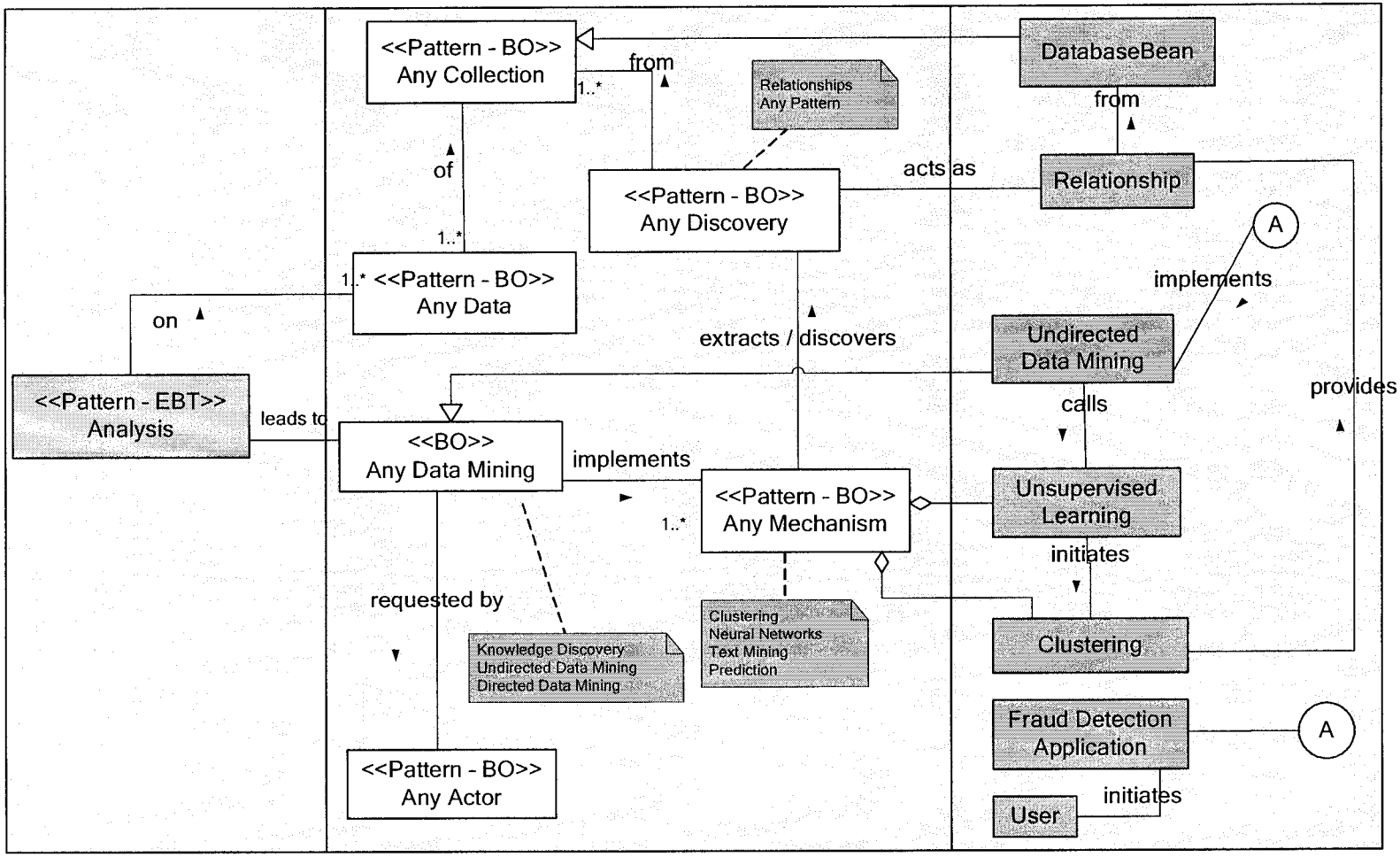


Figure B-4 Credit Card Fraud Detection Application Using Any Data Mining Pattern

B.3.2 Behavior Diagram

Figure B-5 represents the sequence diagram for the use case provided for application 2. This sequence diagram describes the logical flow between the EBTs, BOs and IOs.

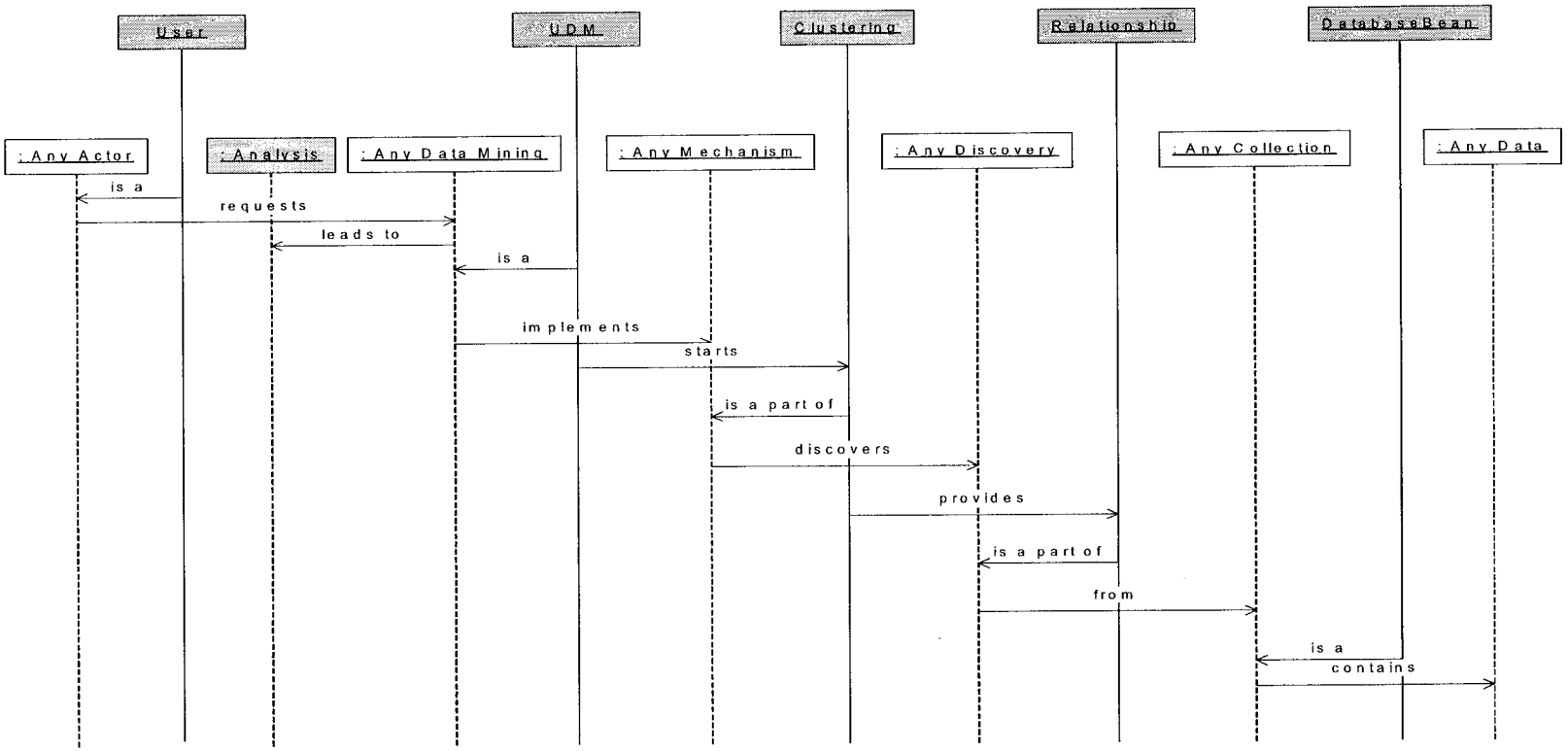


Figure B-5 Sequence Diagram for Application

APPENDIX C Implementation and Code

C.1 Implementation Details

This appendix provides the implementation details for the patterns implemented as a part of this thesis. The patterns implemented are Discovery Stable Analysis Pattern and Any Data Mining Stable Design Pattern. These patterns are implemented up to second level.

C.1.1 Discovery Stable Analysis Pattern

The model for Discovery pattern is represented in Figure C-1. The detailed pattern description is provided in Appendix A.

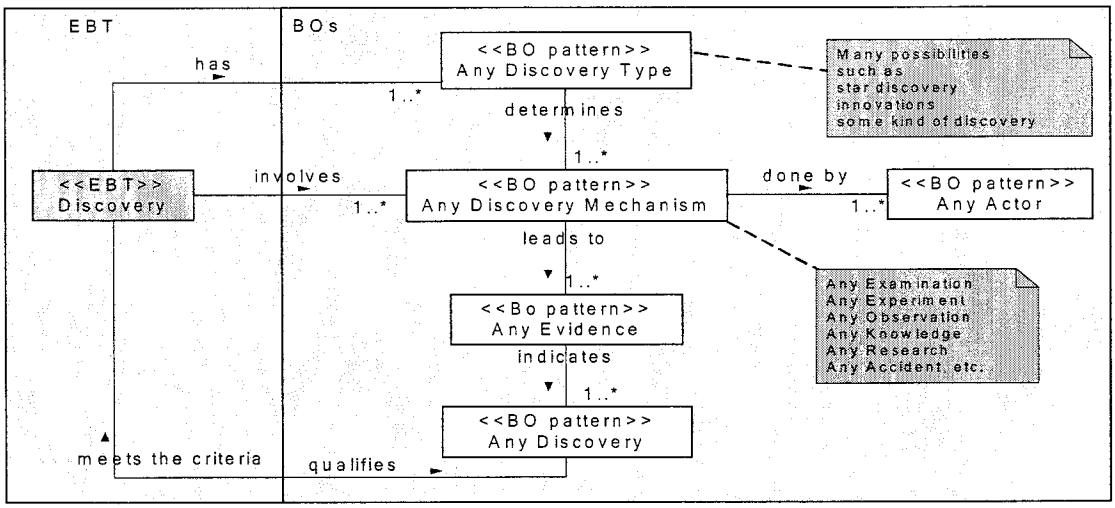


Figure C-1 Discovery Stable Analysis Pattern

Figure C-2 represents the applicability of Discovery Stable Analysis Pattern in knowledge discovery application. The description of Figure C-2 is provided in Appendix A.

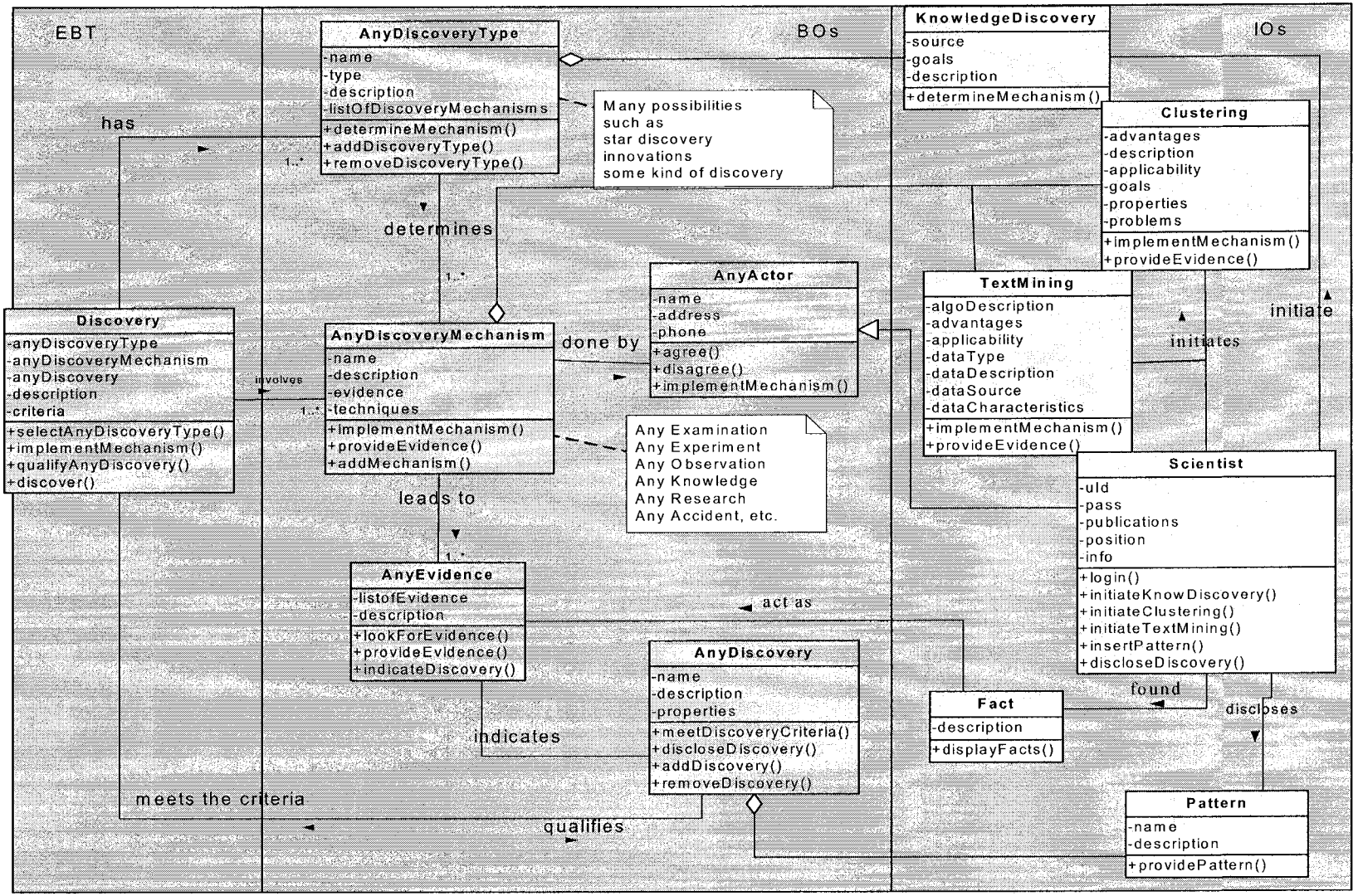


Figure C-2 Discovery Stable Analysis Pattern in Knowledge Discovery Application

C.1.2 Unique Segment of Code

Code for EBT Discovery

Discovery Stable Analysis Pattern is implemented up to two levels. Provided below is the code for the first level patterns.

```
/*
 * Created on May 26, 2005
 *
 * TODO To change the template for this generated file go to
 * Window - Preferences - Java - Code Style - Code Templates
 */

/**
 * @author Pranali Khadpe
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 */

public class Discovery
{

    private String description = "";
    private String criteria ="";

    public Discovery()
    {
            description = "";
            criteria= "";
    }
    /**
     * @param description
     * @param criteria
     */
    public Discovery(String description, String criteria)
    {
            this.description = description;
            this.criteria = criteria;
    }
    /**
     * @return Returns the description.
```

```
 */
public String getDescription()
{
	return description;
}
/**
 * @param description The description to set.
 */
public void setDescription(String description)
{
	this.description = description;
}
/**
 * @return Returns the criteria.
 */
public String getcriteria()
{
	return criteria;
}
/**
 * @param criteria The criteria to set.
 */
public void setcriteria(String criteria)
{
	this.criteria = criteria;
}
/* (non-Javadoc)
 * @see java.lang.Object#toString()
 */
public String toString()
{
	return description + "  " + criteria;
}
public AnyDiscoveryType selectAnyDiscoveryType()
{
	AnyDiscoveryType type = new AnyDiscoveryType();
	type.classifyDiscovery();
	return type;
}
public void implementDiscoveryMechanism()
{
	AnyDiscoveryMechanism mech = new AnyDiscoveryMechanism();
	mech.implementMechanism();
}
```

```
    public boolean qualifyAnyDiscovery()
    {
            AnyDiscovery disc = new AnyDiscovery();
            boolean flag = false;
            if (disc.getProperties().equalsIgnoreCase(criteria))
            {
                    flag = true;
            }
            else flag = false;
            return flag;

    }
    public void examineFeature()
    {
            AnyFeature feature = new AnyFeature();
            String featureName = feature.getName();
    }

}
```

Code for BOs

Any Discovery Pattern

The model for Discovery pattern is expanded to second level. The second level patterns are represented in the following figure. Figure C-3 represents the second level for Any Discovery BO.

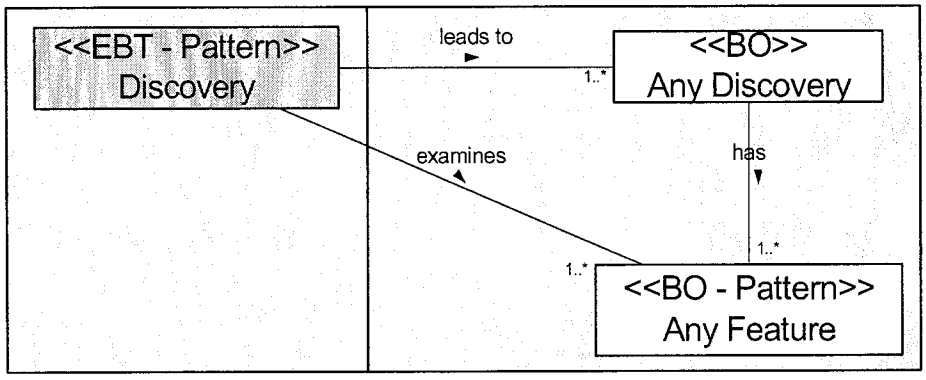


Figure C-3 Second Level Pattern for Any Discovery Pattern

Pattern Description

Discovery (Discovery) leads to one to many Discoveries (Any Discovery), which has one to many features (Any Feature), which are examined by Discovery pattern.

Any Discovery Class

```
/*
 * Created on May 27, 2005
 *
 * TODO To change the template for this generated file go to
 * Window - Preferences - Java - Code Style - Code Templates
 */

/**
 * @author Pranali Khadpe
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 */

public class AnyDiscovery
{
	private AnyFeature feature  = new AnyFeature();
	private String name ="";
	private String description ="";
	private String properties = "";

	/**
	 * @return Returns the description.
	 */
	public String getDescription()
	{
		return description;
	}
	/**
	 * @param description The description to set.
	 */
	public void setDescription(String description)
	{
		this.description = description;
```

```
}
/**
 * @return Returns the name.
 */
public String getName()
{
	return name;
}
/**
 * @param name The name to set.
 */
public void setName(String name)
{
	this.name = name;
}

/**
 * @return Returns the feature.
 */
public AnyFeature getFeature() {
	return feature;
}
/**
 * @param feature The feature to set.
 */
public void setFeature(AnyFeature feature) {
	this.feature = feature;
}
/**
 * @return Returns the properties.
 */
public String getProperties() {
	return properties;
}
/**
 * @param properties The properties to set.
 */
public void setProperties(String properties) {
	this.properties = properties;
}
/* (non-Javadoc)
 * @see java.lang.Object#toString()
 */
public String toString() {
```

```
        return "Name " + name + " Description" + description;
    }

    public boolean meetDiscoveryCriteria()
    {
        boolean flag = false;
        Discovery disc = new Discovery();
    if (properties.equalsIgnoreCase(disc.getcriteria()))
    {
        flag = true;
    }
    else flag = false;
    return flag;
    }
    public String discloseDiscovery()
    {
        return toString();
    }
    public void addDiscovery()
    {
    // done using the hooks
    }
    public void removeDiscovery()
    {
    // done using the hooks
    }
    public void hasFeatures(String name)
    {
        AnyFeature feature = new AnyFeature();
        feature.setName(name);
    }

}
```

Any Discovery Type or Any Type

The pattern Any Discovery Type is the specific version of Any Type. The second level pattern for Any Type is provided in Figure C-4 and the code for first level is provided.

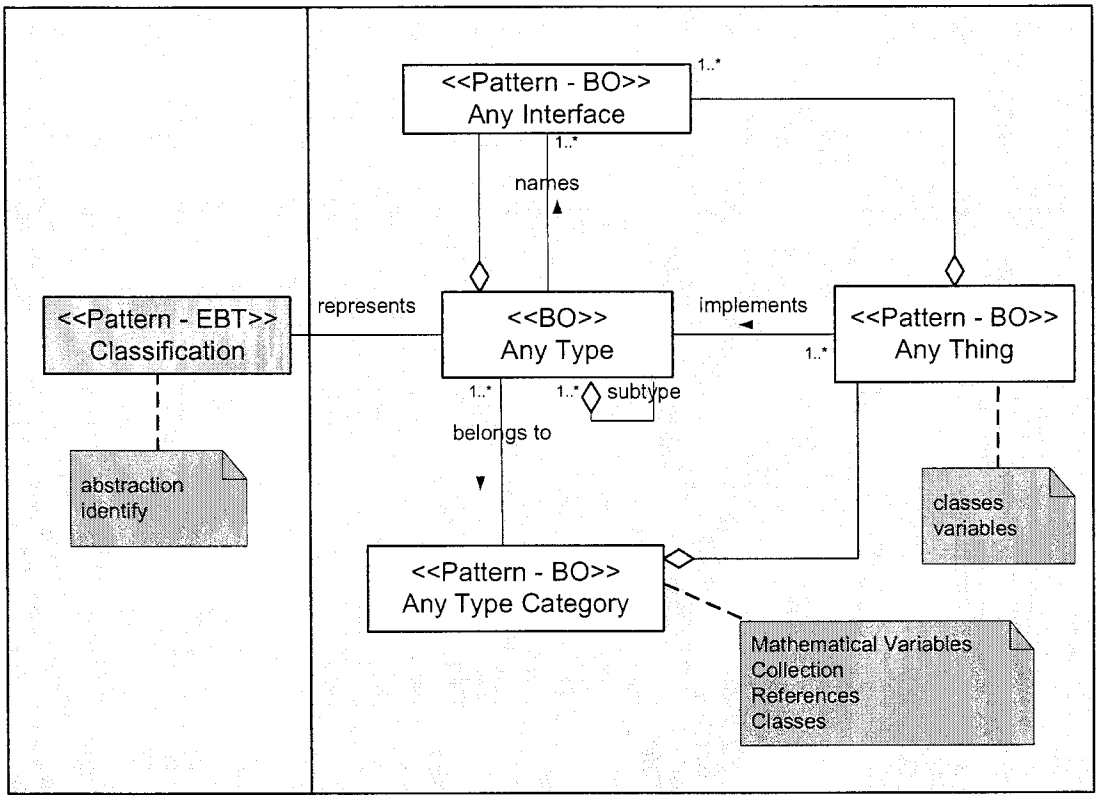


Figure C-4 Second Level Pattern for Any Type

```
/*
 * Created on May 27, 2005
 *
 * TODO To change the template for this generated file go to
 * Window - Preferences - Java - Code Style - Code Templates
 */

/**
 * @author Pranali Khadpe
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
```

```
 */

import java.util.*;
public class AnyDiscoveryType
{
	private String description ="";
	private String name = "";
	private int type = 0;
	private Vector listOfMechanism = null;
	private AnyInterface anyInt = new AnyInterface()


	/**
	 * @return Returns the name.
	 */
	public String getName() {
		return name;
	}
	/**
	 * @param name The name to set.
	 */
	public void setName(String name) {
		this.name = name;
	}
	/**
	 * @return Returns the description.
	 */
	public String getDescription() {
		return description;
	}
	/**
	 * @param description The description to set.
	 */
	public void setDescription(String description) {
		this.description = description;
	}
	/**
	 * @return Returns the type.
	 */
	public int getType() {
		return type;
	}
	/**
	 * @param type The type to set.
```

```
	 */
	public void setType(int type) {
		this.type = type;
	}
	/* (non-Javadoc)
	 * @see java.lang.Object#toString()
	 */
	public String toString()
	{
		// TODO Auto-generated method stub
		String info = "		DiscoveryType:		" + name + "	Type:	" +
type + "		Description:	" + description;
		return info;
	}

	public boolean determineMechanism()
	{
		boolean flag = true;
		AnyDiscoveryMechanism mech = new AnyDiscoveryMechanism();
		return flag;
	}
	public void addDiscoveryType(String mechanism)
	{
		listOfMechanism.addElement(mechanism);
	}
	public void removeDiscoveryType(String mechanism)
	{
		listOfMechanism.removeElement(mechanism);
	}
	public Discovery classifyDiscovery()
	{
		Discovery discovery = new Discovery();
		return discovery;
	}
	public Vector belongTo()
	{
		AnyTypeCategory atc = new AnyTypeCategory();
		atc.catergorize();
		return atc.getCategory();
	}
	public void nameInterface(String name)
	{
		AnyInterface anyInterface = new AnyInterface();
		anyInterface.setName(name);
```

}

}

Any Discovery Mechanism or Any Service

Figure C-5 represents the second level pattern for Any Service pattern. Any Discovery Mechanism is the same as Any Service, as Any Discovery Mechanism is specified version of Any Service. The code for the first level pattern follows the Figure C-5.

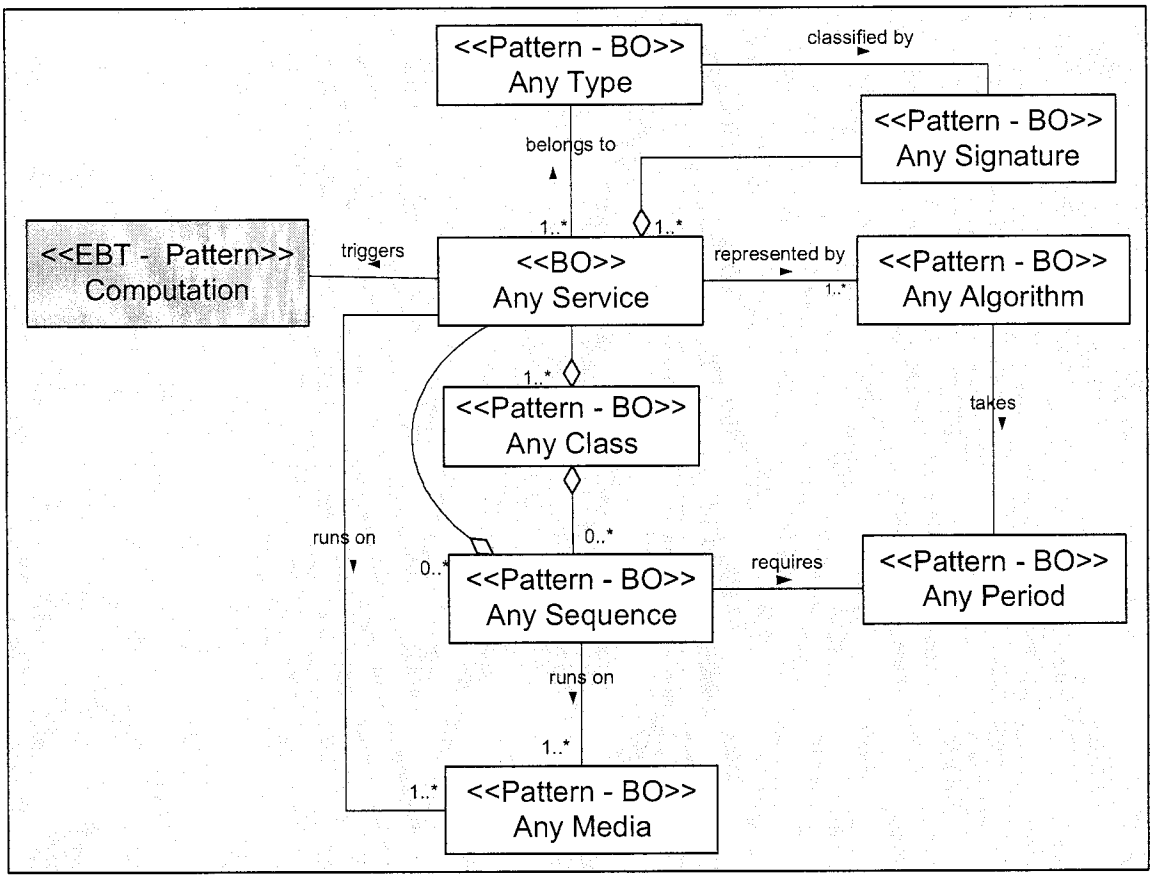


Figure C-5 Second Level Pattern for Any Discovery Mechanism or Any Service.

Any Discovery Mechanism

```
/*
 * Created on May 27, 2005
 *
 * TODO To change the template for this generated file go to
 * Window - Preferences - Java - Code Style - Code Templates
 */
import java.util.*;

/**
 * @author Pranali Khadpe
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 */

public class AnyDiscoveryMechanism
{
	private AnySignature anysig = new AnySignature();
	private String description = "";
	private String name = "";
	private Vector techniques = new Vector();
	/**
	 * @return Returns the name.
	 */
	public String getName() {
		return name;
	}
	/**
	 * @param name The name to set.
	 */
	public void setName(String name) {
		this.name = name;
	}

	/**
	 * @return Returns the description.
	 */
	public String getDescription()
	{
		return description;
	}
	/**
```

```
 * @param description The description to set.
 */
public void setDescription(String description)
{
	this.description = description;
}
/**
 * @return Returns the techniques.
 */
public Vector getTechniques()
{
	return techniques;
}
/**
 * @param techniques The techniques to set.
 */
public void setTechniques(Vector techniques)
{
	//this.techniques = techniques;
}
/* (non-Javadoc)
 * @see java.lang.Object#toString()
 */
public String toString()
{
	// TODO Auto-generated method stub
	return "Name" + name + "Description" + description + "List of
	techniques" + techniques;
}

public void addMechanism(String mech)
{
	techniques.addElement(mech);
}
public void implementMechanism()
{

}
public String provideEvidence(AnyEvidence e)
{
	return e.toString();
}
public AnyActor doneBy()
{
```

```
            AnyActor role = new AnyActor();
            return role;
        }
        public AnyType belongTo()
        {
            AnyType type = new AnyType();
            return type;
        }
        public AnyPeriod timeRequired()
        {
            AnyPeriod period = new AnyPeriod();
            return period;
        }
        public AnyMedia runsOn()
        {
            AnyMedia media = new AnyMedia();
            media.toString();
            return media;
        }
}
```

Any Evidence

Figure C-6 represents the second level pattern for Any Evidence pattern. The code for the first level pattern follows the Figure C-6.

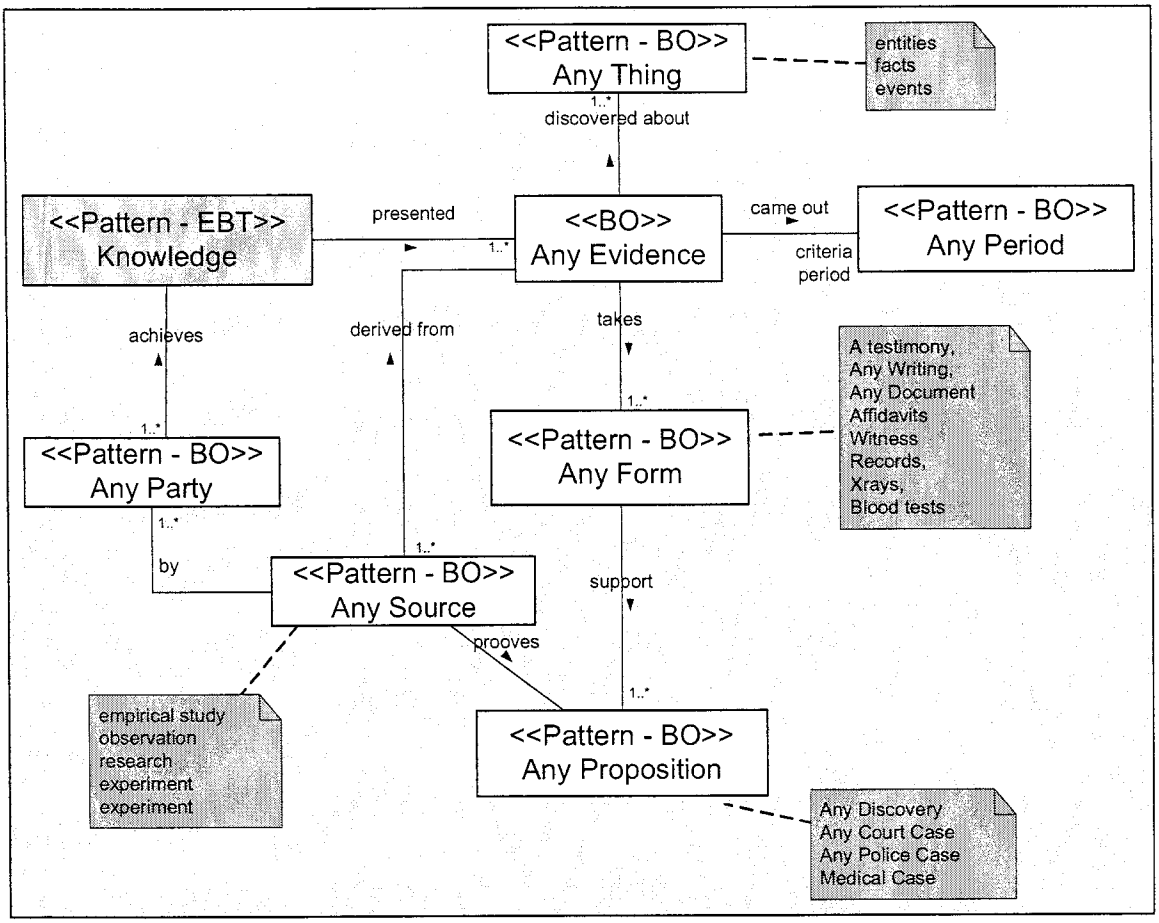


Figure C-6 Second Level Pattern for Any Evidence

Any Evidence

```
/*
 * Created on May 27, 2005
 *
 * TODO To change the template for this generated file go to
 * Window - Preferences - Java - Code Style - Code Templates
 */

/**
 * @author Pranali Khadpe
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 */
```

```
import java.util.Date;
public class AnyEvidence
{

	private String listOfEvidence ="";
	private String description ="";


	/**
	 * @return Returns the description.
	 */
	public String getDescription() {
		return description;
	}
	/**
	 * @param description The description to set.
	 */
	public void setDescription(String description) {
		this.description = description;
	}
	/**
	 * @return Returns the listOfEvidence.
	 */
	public String getListOfEvidence() {
		return listOfEvidence;
	}
	/**
	 * @param listOfEvidence The listOfEvidence to set.
	 */
	public void setListOfEvidence(String listOfEvidence) {
		this.listOfEvidence = listOfEvidence;
	}

	/* (non-Javadoc)
	 * @see java.lang.Object#toString()
	 */
	public String toString() {
		// TODO Auto-generated method stub
		return "Description     :" + description + "     ListOfEvidence: " +
listOfEvidence;
	}
	public void lookForEvidence()
	{
	}
```

```
        public AnyEvidence provideEvidence()
        {
                return this;
        }
        public AnyDiscovery indicateDiscovery()
        {
                AnyDiscovery disc = new AnyDiscovery();
                disc.toString();
                return disc;
        }
        public AnyThing discoverAnyThing()
        {
                AnyThing thing = new AnyThing();
                thing.toString();
                return thing;
        }
        public String derivedFrom()
        {
                AnySource source = new AnySource();
                return source.getLocation();
        }

        public void takeForms()
        {
                AnyForm form = new AnyForm();
                form.takeForm();
        }
        public Date cameOut()
        {
                AnyPeriod period = new AnyPeriod();
                return period.provideDate();
        }
        public void prooveProposition()
        {

        }
}
```

Any Actor

Figure C-7 represents the second level pattern for Any Actor pattern. The code for the first level pattern follows the Figure C-7.

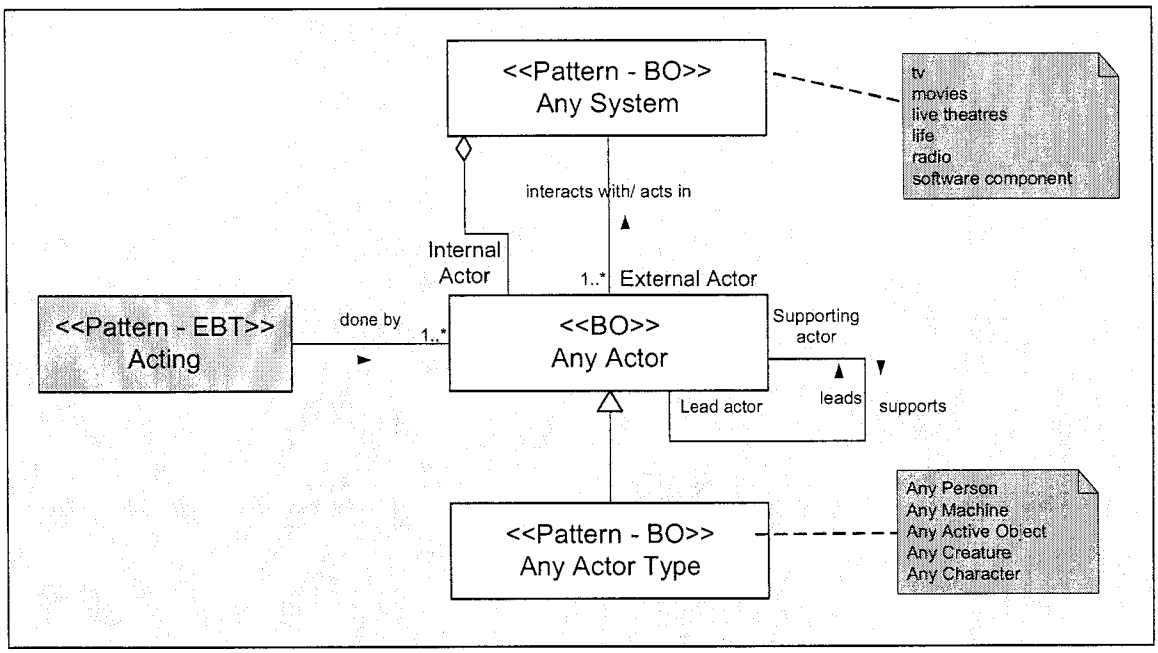


Figure C-7 Second Level Pattern for Any Actor Pattern

```
/*
 * Created on Jun 15, 2005
 *
 * TODO To change the template for this generated file go to
 * Window - Preferences - Java - Code Style - Code Templates
 */

/**
 * @author Pranali Khadpe
 *
 * TODO To change the template for this generated type comment go to
 * Window - Preferences - Java - Code Style - Code Templates
 */
public class AnyActor
{
	private String name;
	private String email;

	public AnyActor() {
		super();
	}
	/**
	 * @return Returns the email.
	 */
```

```
	public String getEmail() {
		return email;
	}
	/**
	 * @param email The email to set.
	 */
	public void setEmail(String email) {
		this.email = email;
	}
	/**
	 * @return Returns the name.
	 */
	public String getName() {
		return name;
	}
	/**
	 * @param name The name to set.
	 */
	public void setName(String name) {
		this.name = name;
	}
	public String toString()
	{
		return "Name  :" + name + "Email     :" + email;
	}
	public void implementMechanism()
	{

	}
	public void interact()
	{
	}

	public void act()
	{

	}
}
```

APPENDIX D Instructions and Demo

D.1 Setup Requirements

This section provides the installation, configuration and deployment guidelines for Java, Tomcat and developed application, respectively.

D.1.1 Installing the Java Development Kit (JDK)

The first step is to download and install Java. For this purpose we require JDK 1.3 or a later version. The http://java.sun.com/j2se/1.4/download.html site provides download and installation information. Once Java is installed, the environment variable PATH should be configured properly. Opening a DOS window and typing "java -version" and "javac -help" can do this.

D.1.2 Configuring Tomcat

The second step is to configure Apache Tomcat. Apache Tomcat can be installed using http://jakarta.apache.org/tomcat/index.html website. Once Tomcat is installed change the port to 80 from 8080. To change the port, edit *install_dir/conf/server.xml* and change the port attribute of the Connector element from 8080 to 80, yielding the result below.

```
<Connector
className="org.apache.coyote.tomcat4.CoyoteConnector"
port="80" ..... />
```

Setup the JAVA_HOME variable. JAVA_HOME environment variable tells Tomcat where to find Java. Failing to properly set this variable prevents Tomcat from handling JSP pages. On Windows NT, or 2000, or XP, you could also go to the Start menu and select Settings, then Control Panel, then System, and then Environment. Then, you could enter the JAVA_HOME value. Test the server: Test some html or jsp pages by going to url *http://localhost*

D.1.3 Deploying the Application

The application is provided in the form of .war files. These war files should directly go under "webapps" folder, which is under root TOMCAT. Tomcat installs and loads the application automatically.

To start discovery of planetary systems go to url http://localhost/servlets-examples/servlet/PLoginServlet

To start knowledge discovery application go to url http://localhost/servlets-examples/login.html

To start credit card fraud detection application go to url http://localhost/datamining/fdetection.jsp

To start moviegoer's application go to url http://localhost/datamining/MarketBasedAnalysis.jsp

D.1.4 Target System

The target system should have the following specifications

1. Windows XP operating system
2. Pentium 4 Processor
3. Minimum 256 MB RAM
4. Minimum 40 GB hard drive.

D.2 Demo Instructions and Snapshots

This section provides the demo instructions and the screen shots of the different form presented in the applications.

D.2.1 Discovery Stable Analysis Pattern - Knowledge Discovery Application

This application uses the example of knowledge discovery to utilize the framework and functionality provided by Discovery pattern.

EBT – Discovery

Table D-1 Overview of BOs and IOs in Knowledge Discovery Application.

BOs	**IOs**
Any Discovery Type	Knowledge Discovery
Any Discovery Mechanism	Clustering, Text Mining
Any Evidence	Fact
Any Actor	Scientist or any user
Any Discovery	Pattern

Knowledge Discovery Application Description

Knowledge discovery is defined as "the non-trivial extraction of implicit, unknown, and potentially useful information from data" [21]. Knowledge discovery process takes the raw results from data mining (the process of extracting trends or patterns from data) or databases and carefully and accurately transforms them into useful and understandable information. This understandable information is presented in the form of patterns. This information is not typically retrievable by standard techniques but is uncovered through the use of Artificial Intelligence (AI) techniques such as Clustering or Text Mining. The results or the facts obtained after using these techniques act as evidence to prove the results. Knowledge discovery is a part of Any Discovery Type and it uses full functionality provided by Any Discovery Type. Also, Any Actor role is played by scientist or any user, who initiates the knowledge discovery, initiates the AI algorithms, finds facts, and discloses results in the form of patterns to the outside world. User is also given the option of recording his discovery.

The application described above is implemented using Core Java, JDBC, Servlets, JSPs, and Apache Tomcat web server. Figures below are the snapshots of the developed application. Figure D-1 is the login page, which illustrates that only the authentic users can use the system and can initiate the discovery process.

Figure D-1 Login.html

After clicking the button Login the LoginServlet is invoked, which validates the user and provides information about knowledge discovery. This servlet also gives the option to the user to start the knowledge discovery process. This is demonstrated in Figure D-2

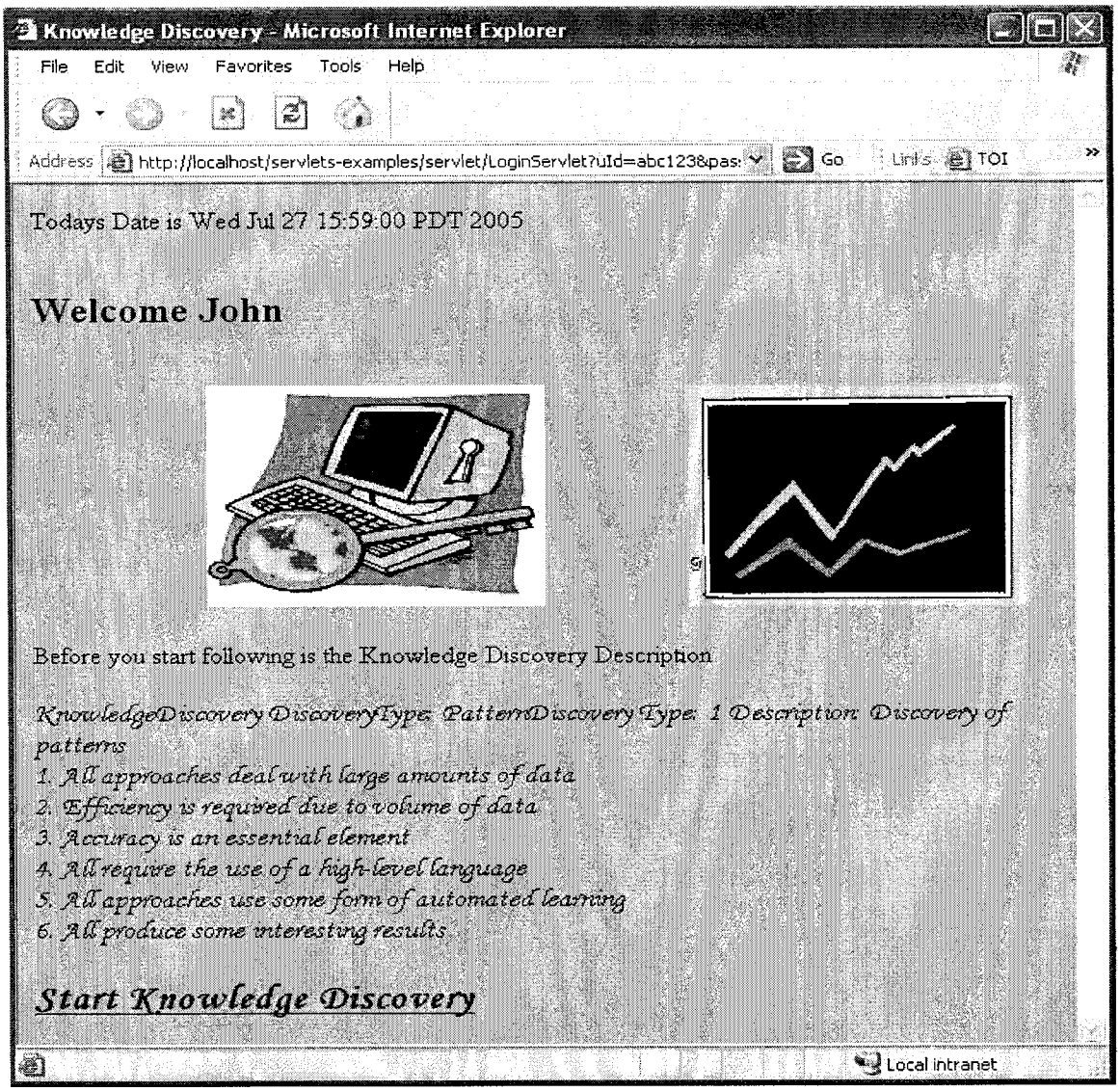


Figure D-2 Knowledge Discovery LoginServlet

After the user clicks on "Start Knowledge Discovery" the user is provided with the option of reading information about the algorithms and then trying the different algorithms. The user can click on the "Information about Algorithms" in Figure D-3 to

read the information. This is represented in Figures D-4 and D-5. Also, the user can start the algorithms without reading the information.

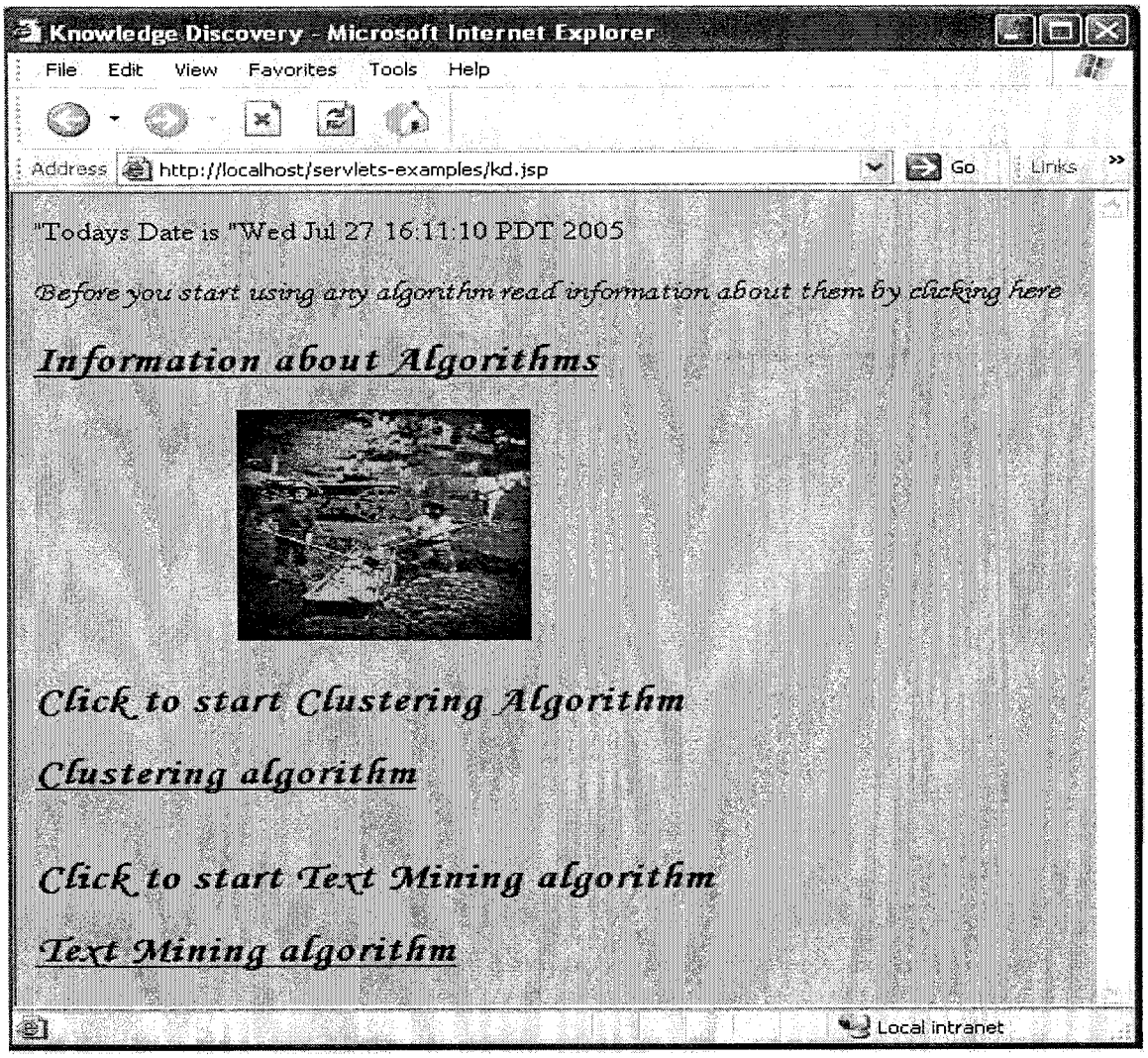


Figure D-3 Knowledge Discovery.jsp

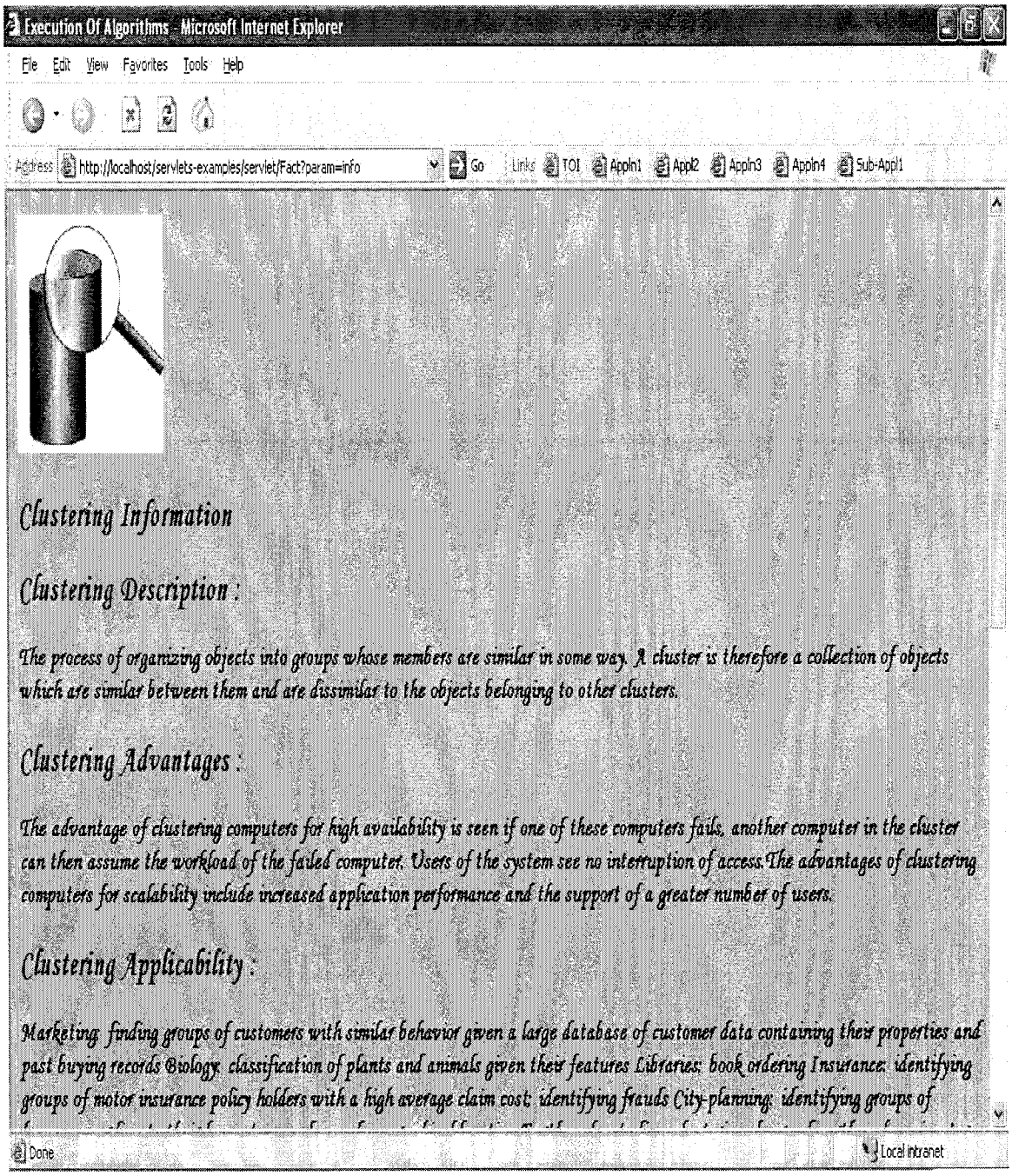


Figure D-4 Information About Algorithms - Clustering

Figure D-5 is continuity of Figure D-4. Figure D-4 shows Clustering information and Figure D-5 shows Text Mining algorithm information. The user is provided the facility to go back and start the algorithm.

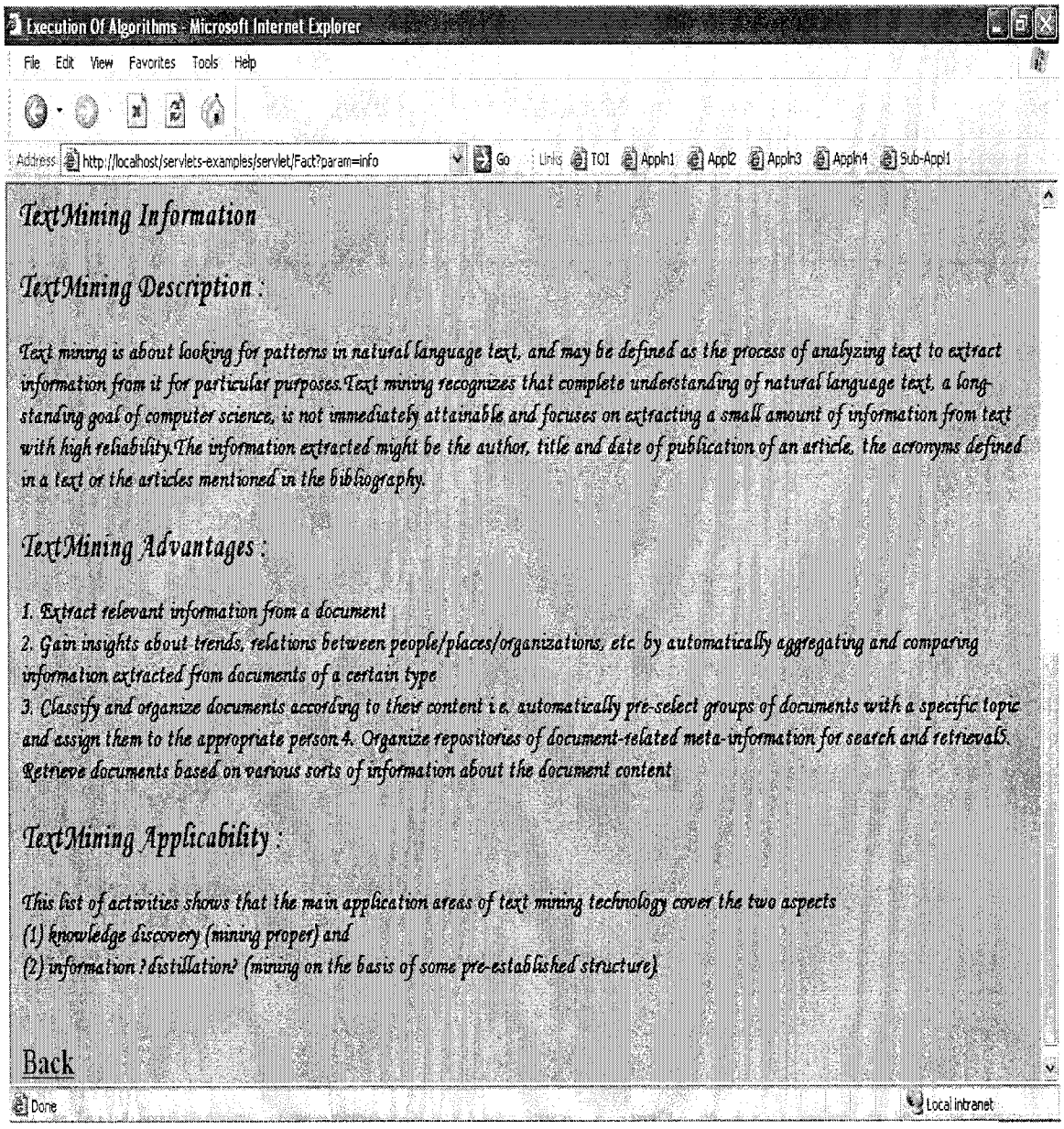


Figure D-5 Information About Algorithms - Text Mining

Figure D-6 shows the results of the implementing the clustering algorithm

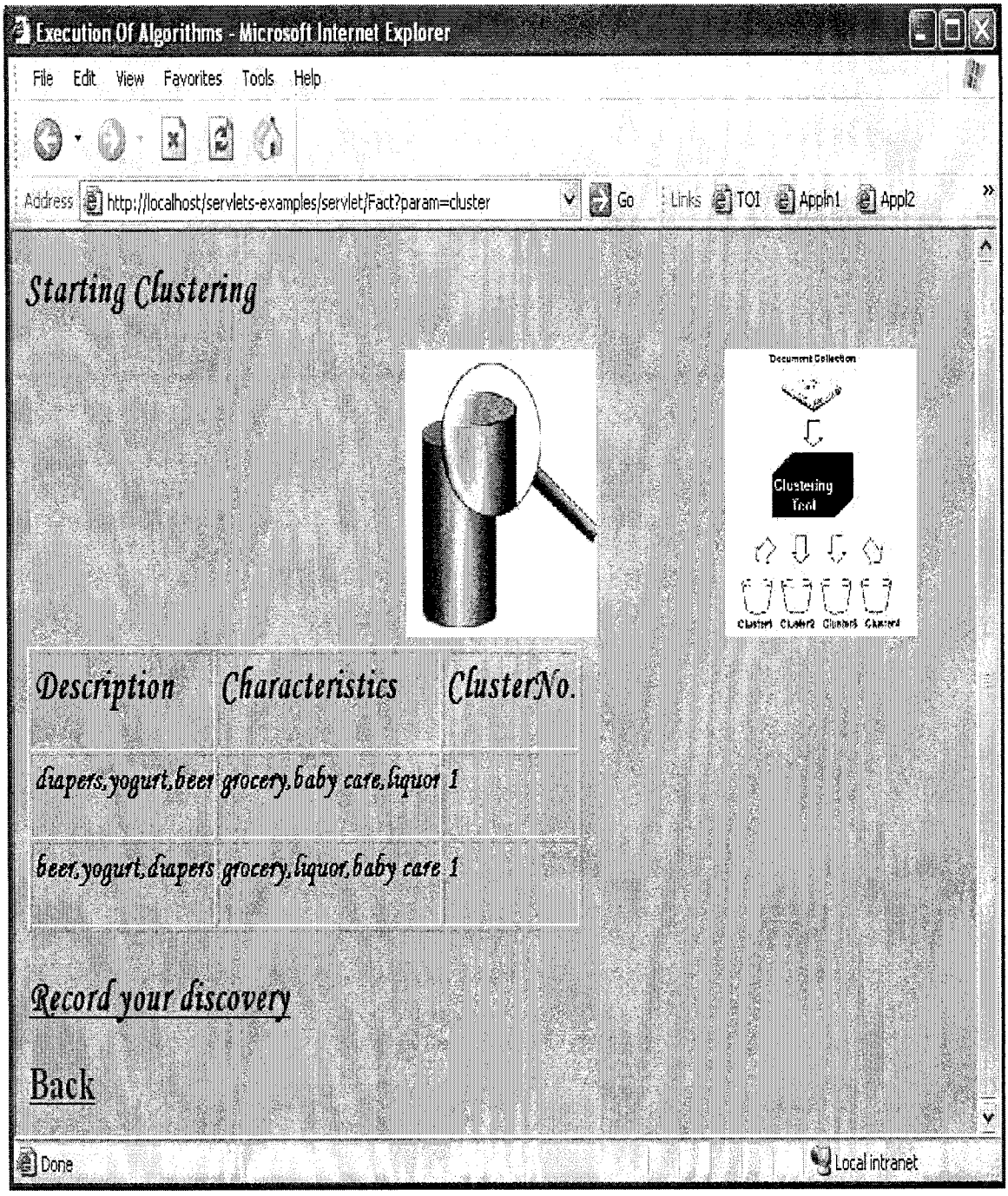


Figure D-6 Implementation of Clustering Algorithm.

Figure D-7 shows results of implementing the Text Mining algorithm.

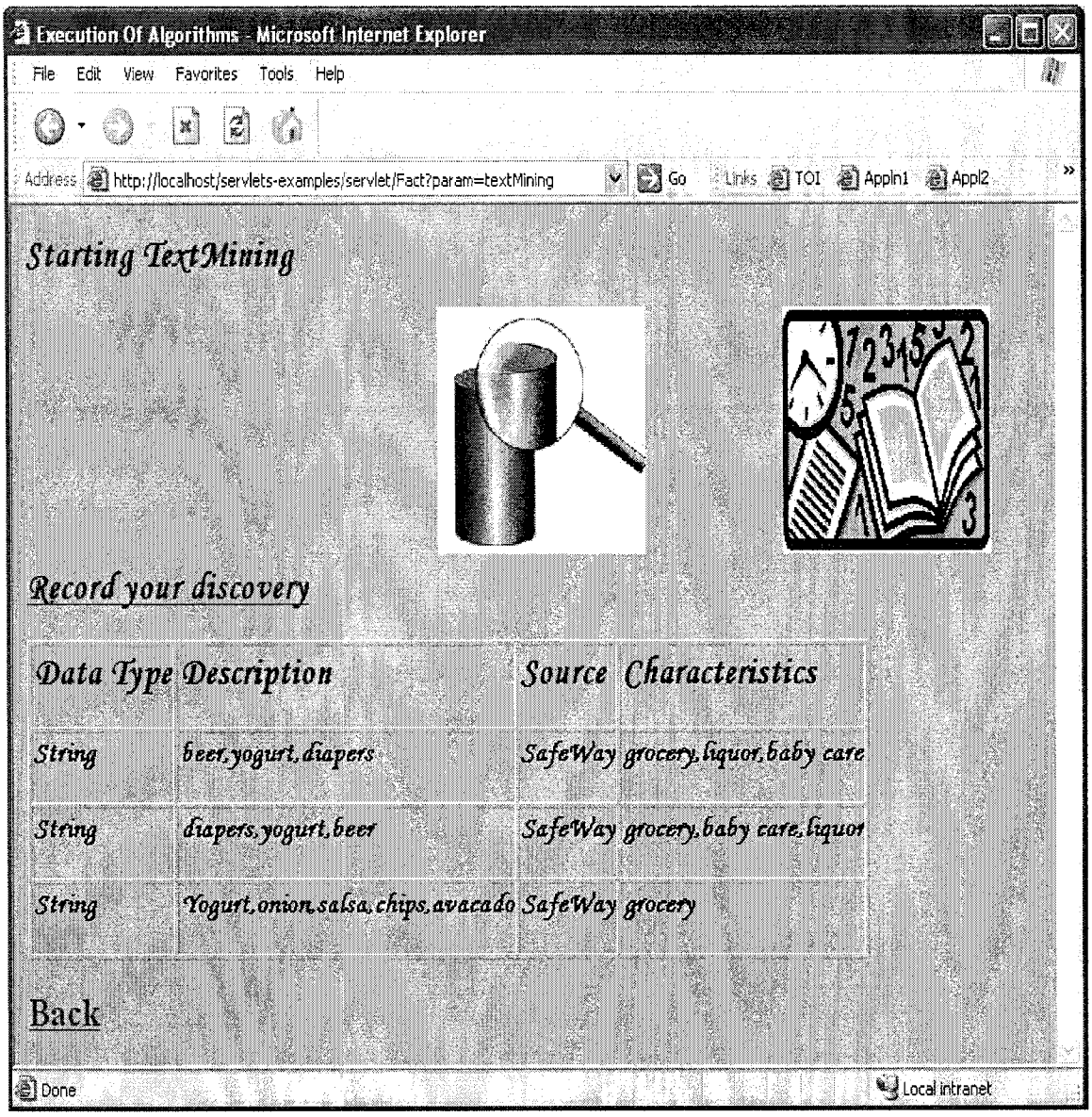


Figure D-7 Implementation of Text Mining Algorithm

The application also provides the feature to record the pattern discovered. This is demonstrated in Figure D-8. The application provides the feature to add the pattern name and the pattern description.

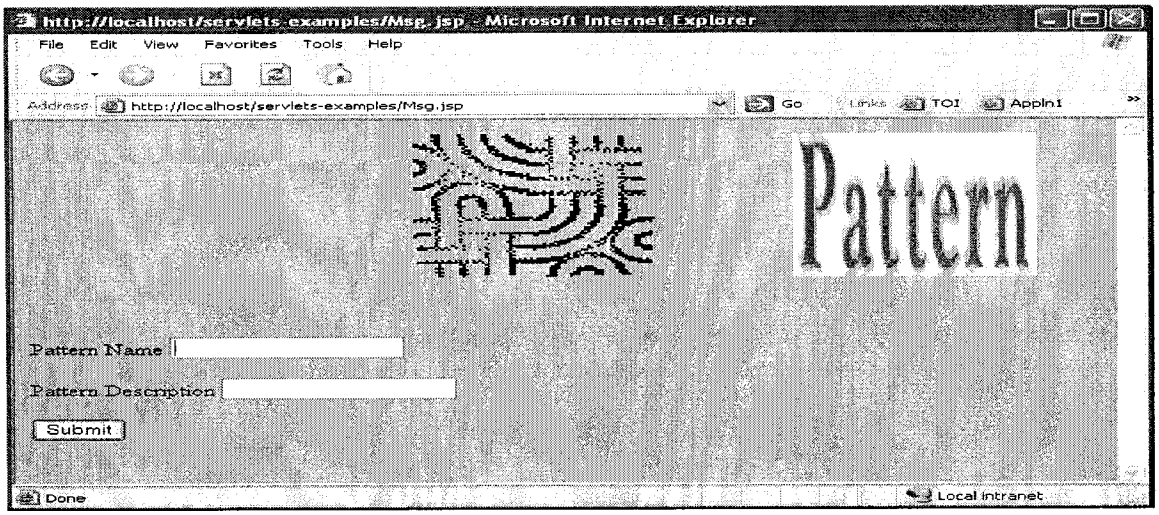


Figure D-8 Pattern Servlet to Record the Pattern

D.2.2 Discovery Stable Analysis Pattern - Planet Discovery Application

This application uses the example of discovery of planet to utilize the framework and functionality provided by Discovery pattern. This application illustrates the diversity of the pattern it shows how Discovery pattern can be used in discovery of a planet. Similar application can be built to demonstrate discovery of a star.

EBT – Discovery

Table D-2 Overview of BOs and its IOs in Discovery of a Planet Application

BOs	**IOs**
Any Discovery Type	Planetary System
Any Discovery Mechanism	Research, Observation
Any Evidence	Fact
Any Actor	Scientist
Any Discovery	Planet

Planet Discovery Application Description

Everyday, scientists conduct research and observe the space to discover new stars or new planets. Over a period of time they have been successful in discovering new stars and new planets. This application demonstrates the discovery of a new planet, which belongs to Pegasi 51 Planetary System. In this application, planetary system is a part of Any Discovery Type, and it determines the discovery mechanisms, which in this case is Research. Research leads to Observation, which acts as evidence. Also, the scientist, who inherits from Any Actor, is responsible for starting the discovery process, initiating the Research, recording the Observation and finally recording the discovery.

Figure D-9 is the first screen the user sees in this application. This screen gives information about the planetary system and gives the user the option to start the discovery process.

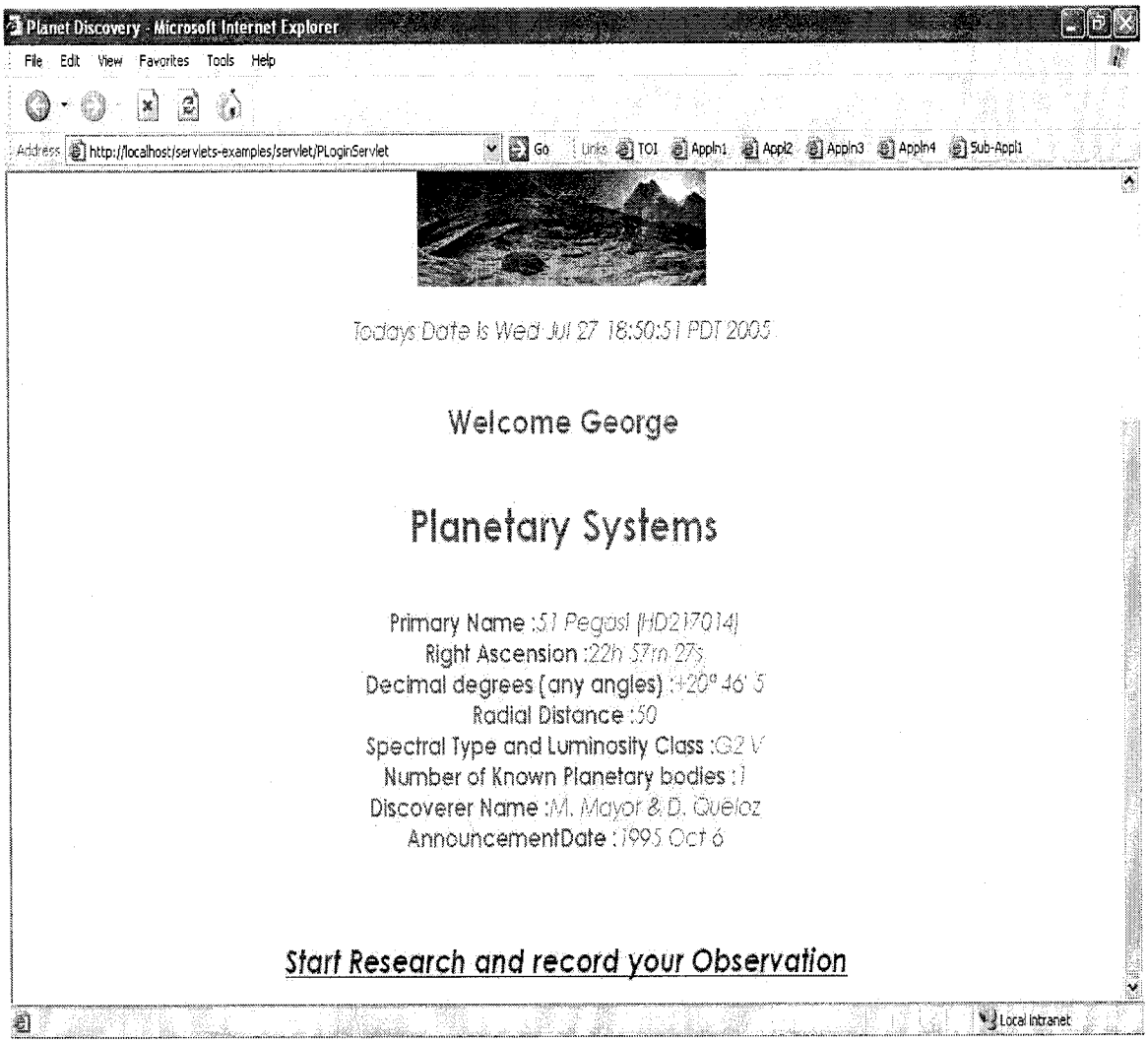


Figure D-9 Planetary System

After the user starts the discovery process, the user is presented with the research information and is asked to record the observation. This is demonstrated in Figure D-10.

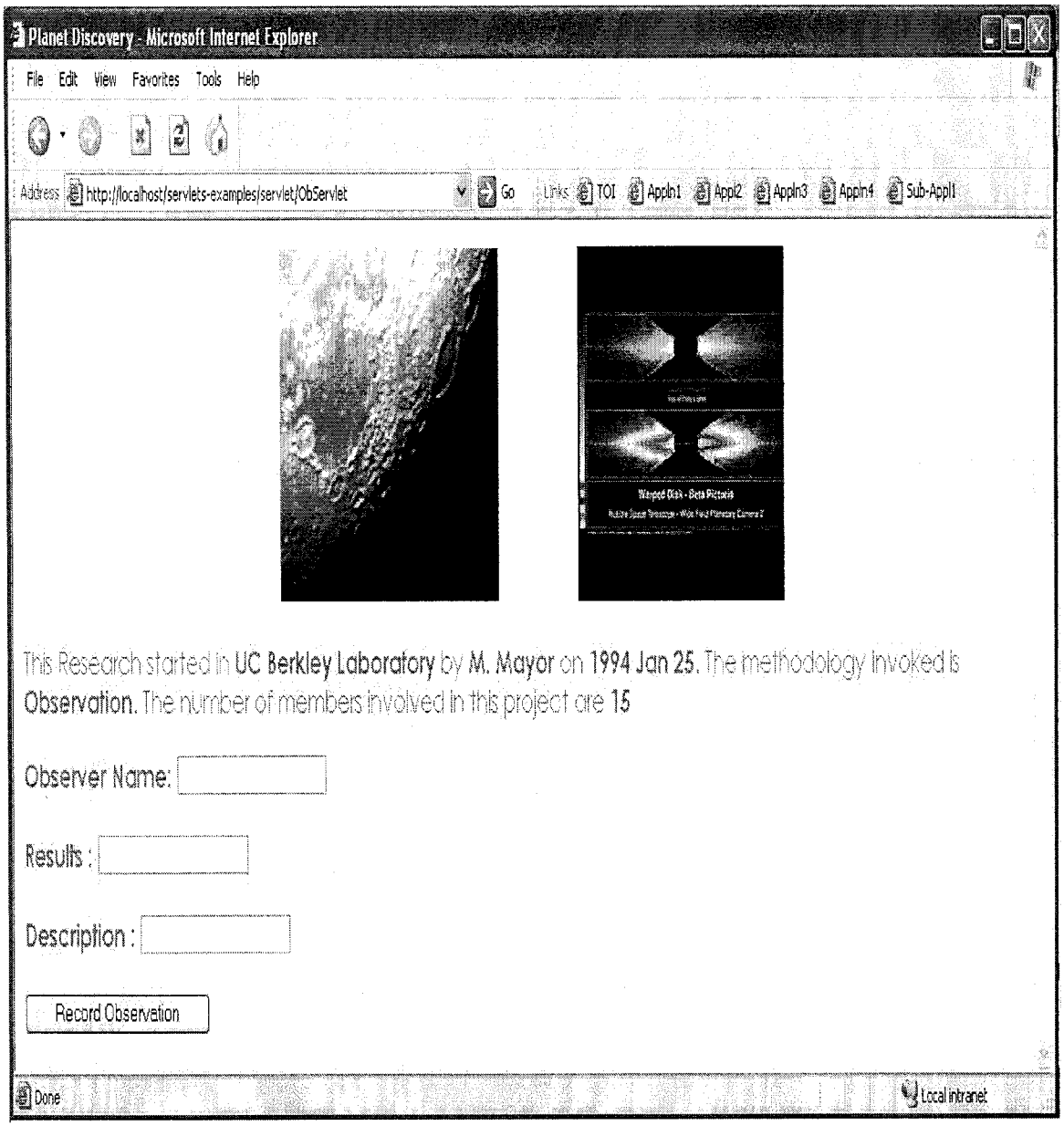


Figure D-10 Observation Servlet

Figure D - 11 shows the different recorded Observations. These recorded Observations consist of the Observation inserted by other members in the past and also the recent Observations.

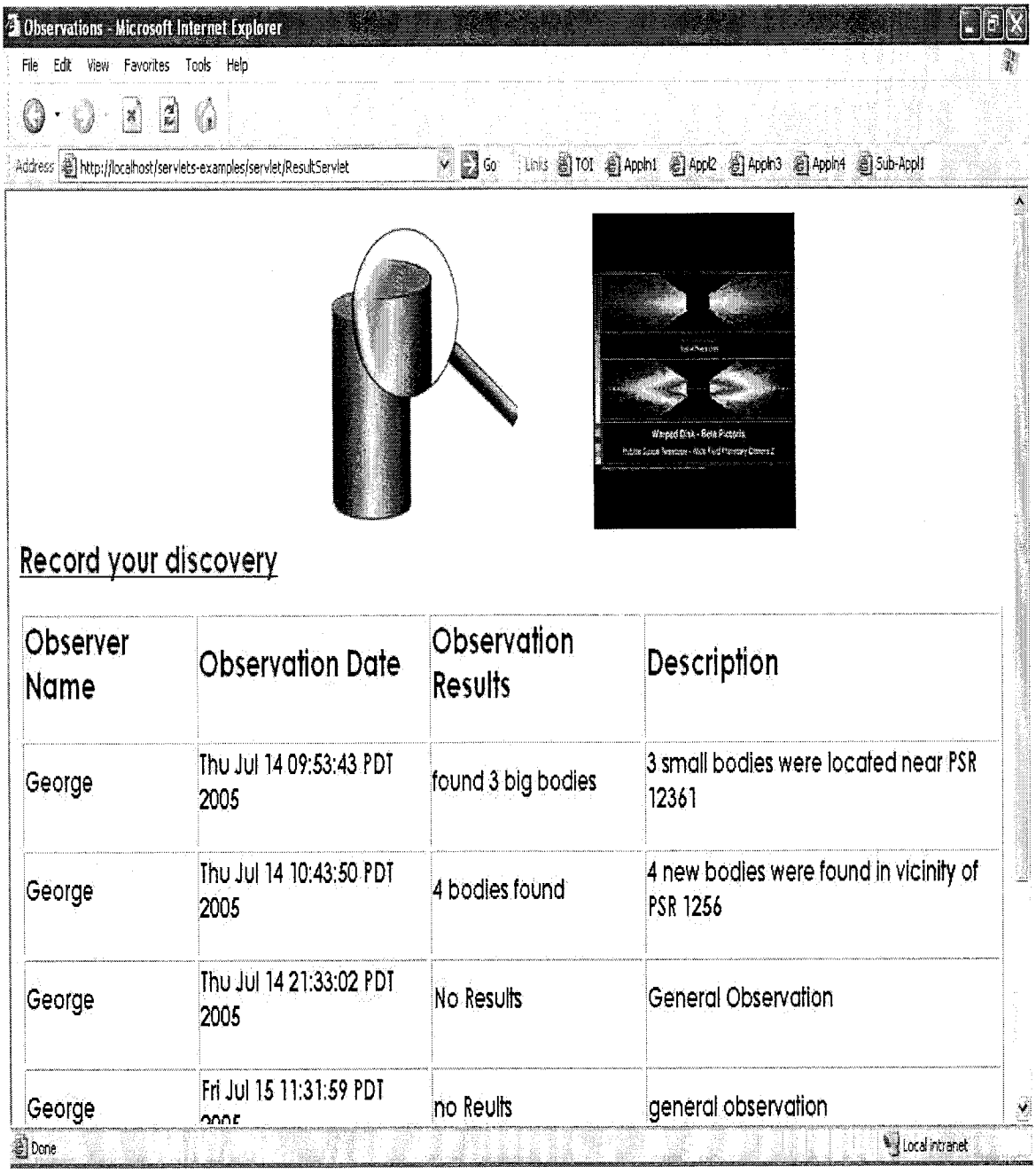


Figure D-11 List of Recorded Observations

The next screen shown in Figure D-12 records the discovery, which in this case is planet. It requires the user to insert all the planet properties.

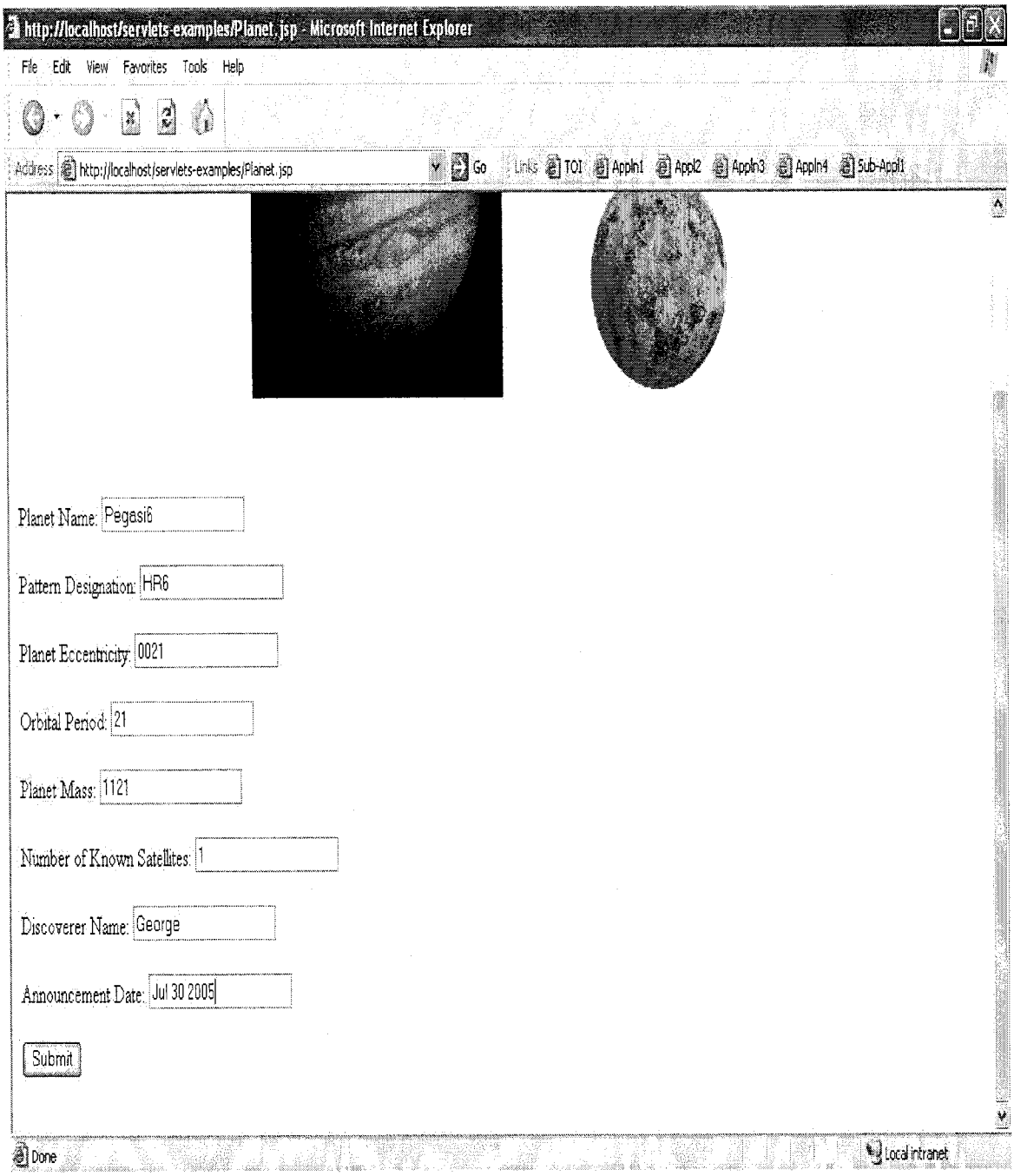
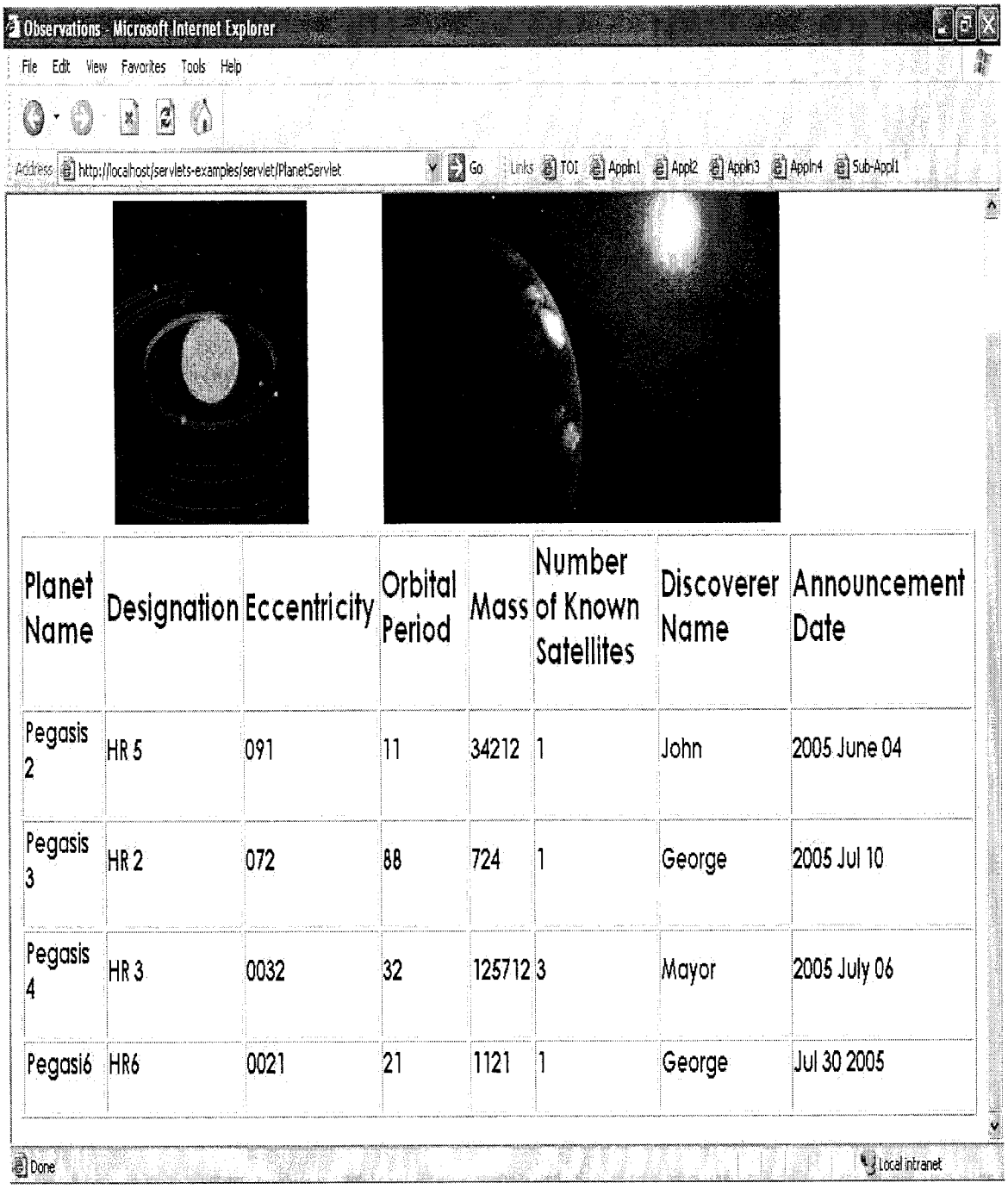


Figure D-12 Planet Recording Screen

Figure D-13 shows all the recorded planets.



Planet Name	Designation	Eccentricity	Orbital Period	Mass	Number of Known Satellites	Discoverer Name	Announcement Date
Pegasis 2	HR 5	091	11	34212	1	John	2005 June 04
Pegasis 3	HR 2	072	88	724	1	George	2005 Jul 10
Pegasis 4	HR 3	0032	32	125712	3	Mayor	2005 July 06
Pegasi6	HR6	0021	21	1121	1	George	Jul 30 2005

Figure D-13 List of Recorded Planets

D.2.3 Any Data Mining Pattern - Credit Card Fraud Detection Application

This application uses the example of credit card fraud detection to utilize the framework and functionality provided by Any Data Mining pattern.

BO – Any Data Mining

Table D-3 Overview of BOs and IOs in Credit Card Fraud Detection Application

BOs	IOs
Any Data Mining	Undirected Data mining
Any Mechanism	Clustering, Unsupervised learning
Any Collection	DatabaseBean
Any Actor	User
Any Discovery	Relationship

Credit Card Fraud Detection Application Description

Credit card fraud detection application uses Undirected Data Mining (UDM) to analyze data. The user is provided with the description, advantages and properties of UDM and is asked to start the data mining process. This is demonstrated in Figure D-14. UDM implements Unsupervised Learning technique, which consists of Clustering mechanism. The user is provided with the description and applicability of Unsupervised Learning so that he can make an intelligent decision. The user is also given the option to begin the Clustering process. This is illustrated in Figure D-15. The user is presented with the results of Clustering and also with information related to Clustering algorithm as shown in Figure D-16. If there is any trigger, the user is provided with the transaction information as shown in Figure D-17.

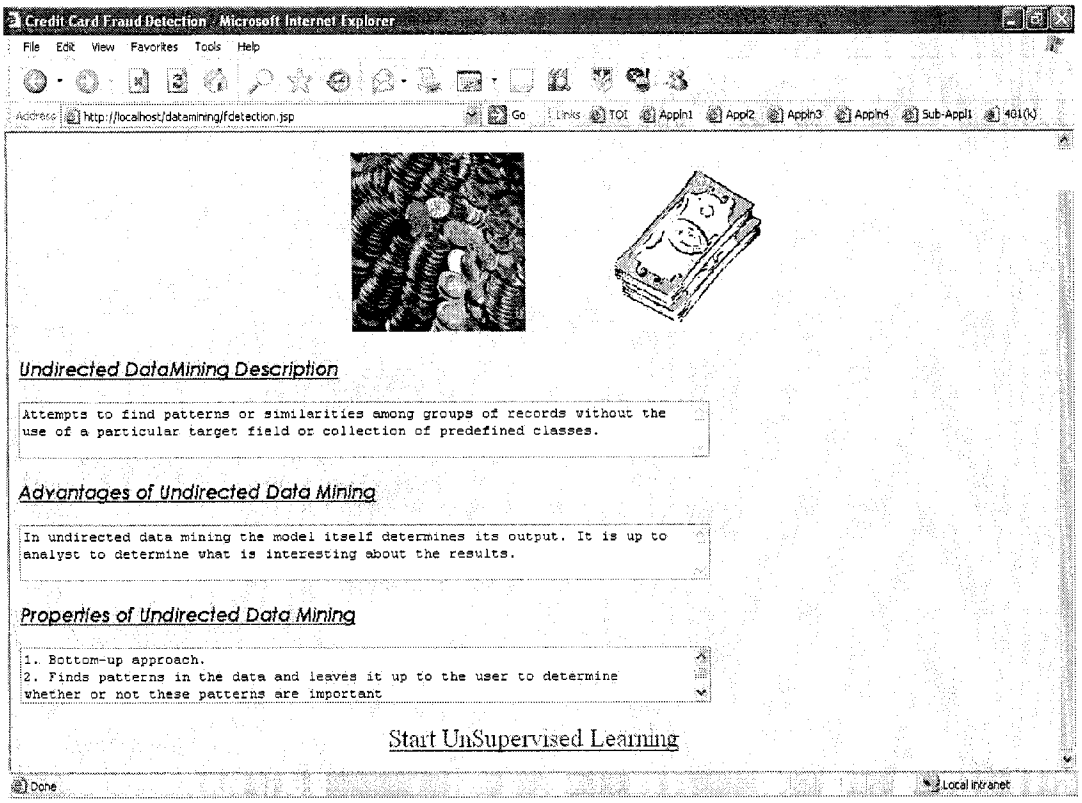


Figure D-14 Fdetection.jsp

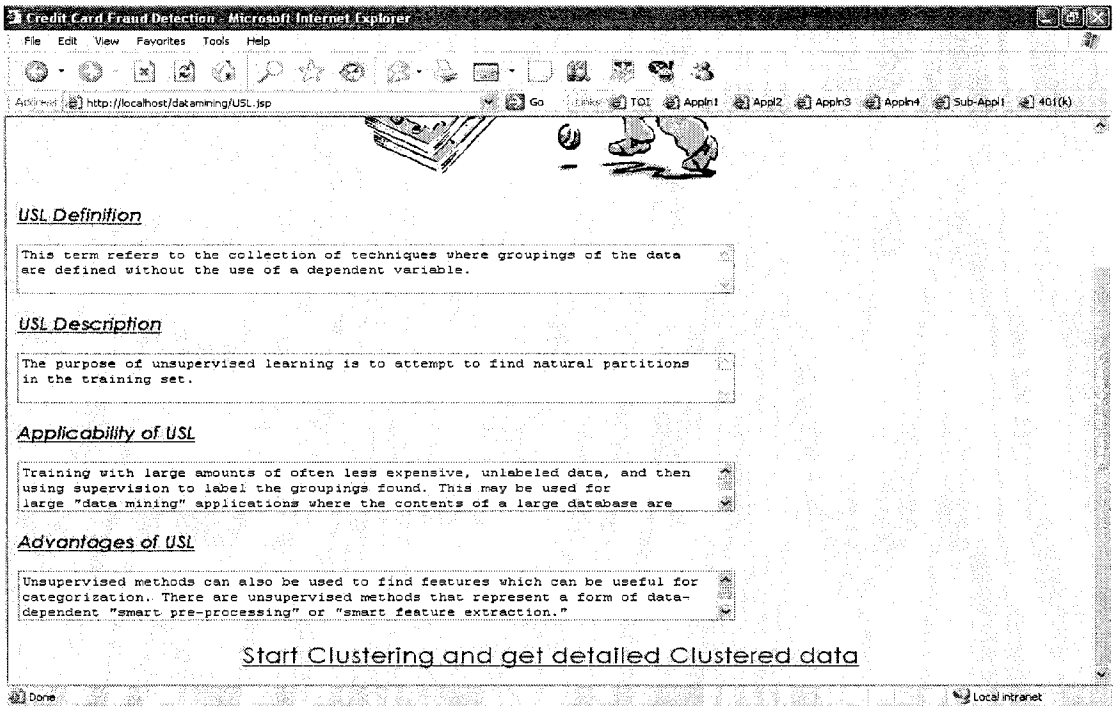


Figure D-15 Unsupervised Learning (usl.jsp)

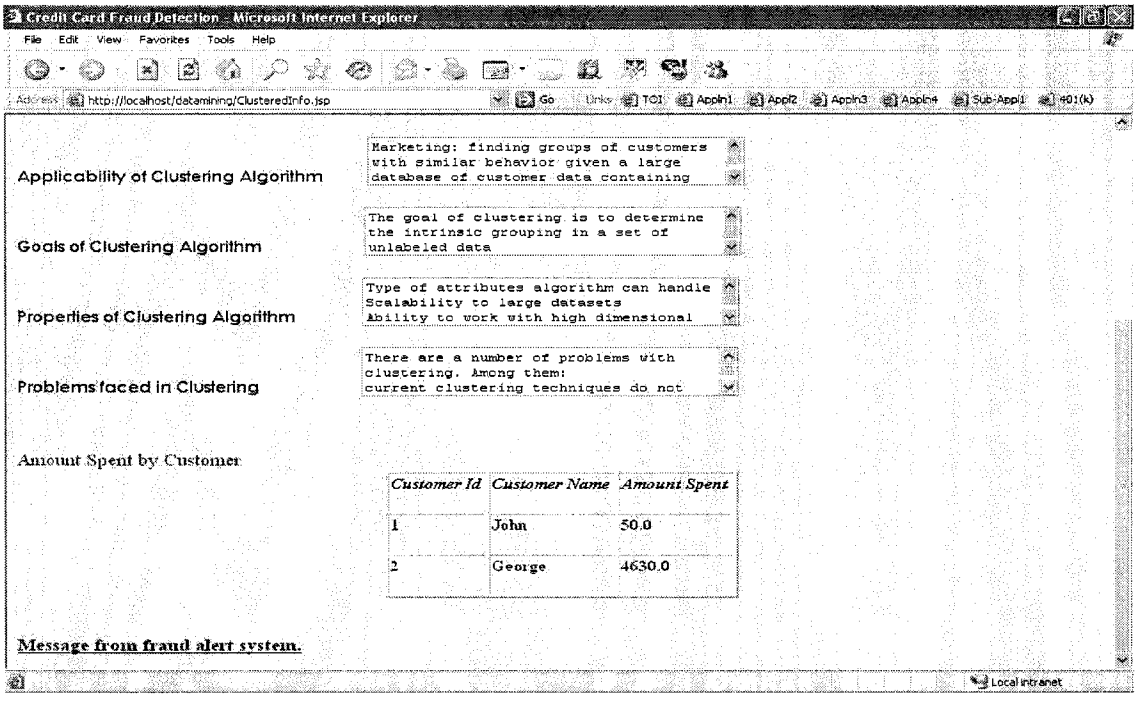


Figure D-16 ClusteredInfo.jsp

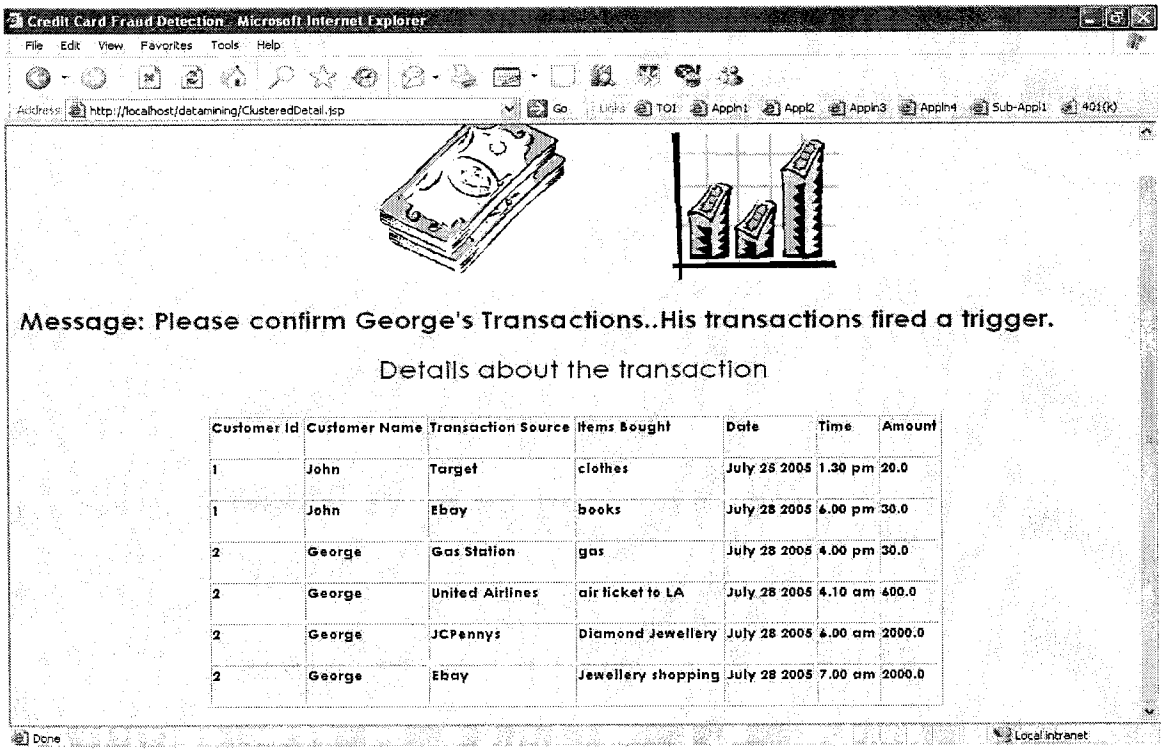


Figure D-17 ClusteredDetail.jsp

D.2.4 Any Data Mining Pattern - Moviegoer's Application

This application uses the example of moviegoer's application to use the framework and functionality provided by Any Data Mining pattern.

Table D-4 Overview of BOs and Corresponding IOs in Moviegoer's Application

BOs	**IOs**
Any Data Mining	Exploratory Data Analysis, Market Based Analysis
Any Mechanism	Prediction, Estimation
Any Collection	DatabaseBean
Any Actor	User
Any Discovery	Result

Moviegoer's Application Description

Moviegoer's application uses two different data mining methodologies, namely Market Based Analysis and Exploratory Data Analysis. Market Based Analysis uses Prediction as the data mining mechanism and Exploratory Data Analysis uses Estimation as the data mining mechanisms. The first screen shown in Figure D-18 provides information about Market Based Analysis and Exploratory Data Analysis so that the user can make a knowledgeable decision. If the user chooses to select Prediction he or she is provided with screen shown in Figure D-19, which provides results. If the user chooses to select Estimation he or she is directed to Estimation.jsp, which is shown in Figure D-20. In this case the user is provided with 3 choices, which allow the user to view the results of the survey taken, or estimation of people according to movie rating, or estimation of number of males or females going for a particular movie. The results of the three choices are demonstrated in Figures D-21, D-22, and D-23.

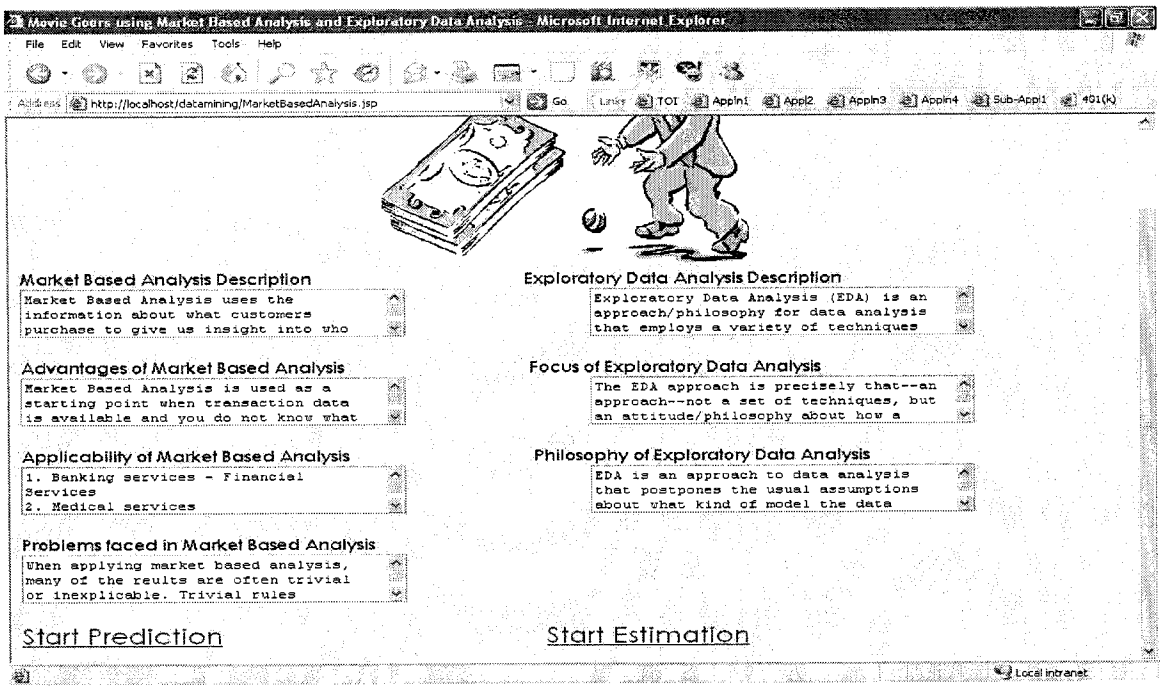


Figure D-18 Moviegoer's Application

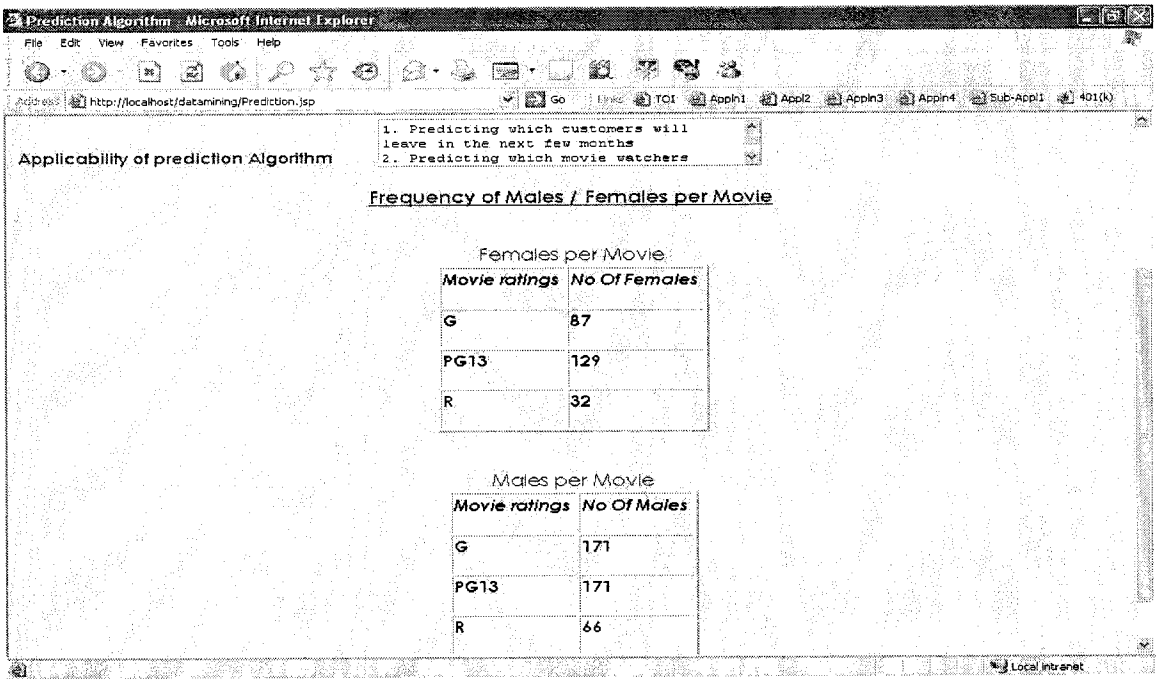


Figure D-19 Prediction.jsp

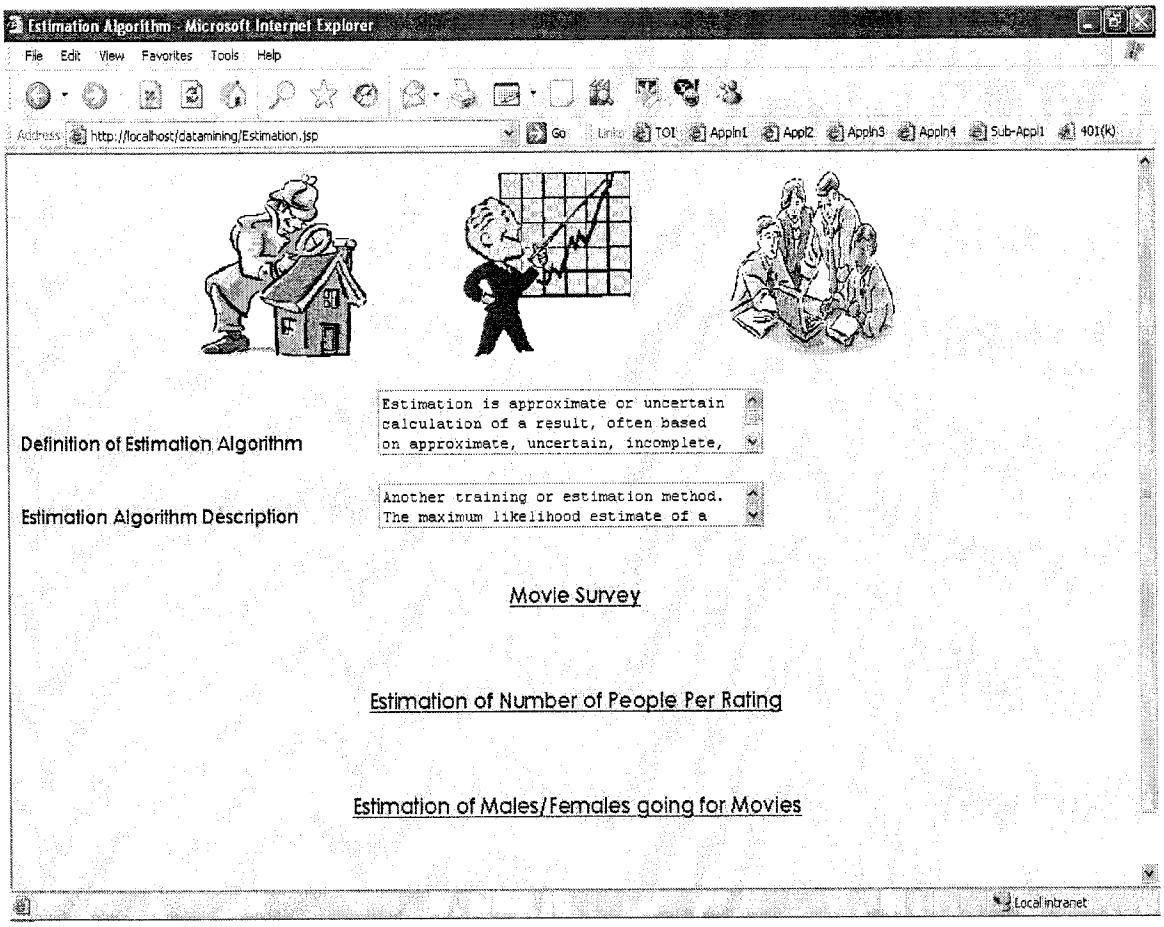


Figure D-20 Estimation.jsp

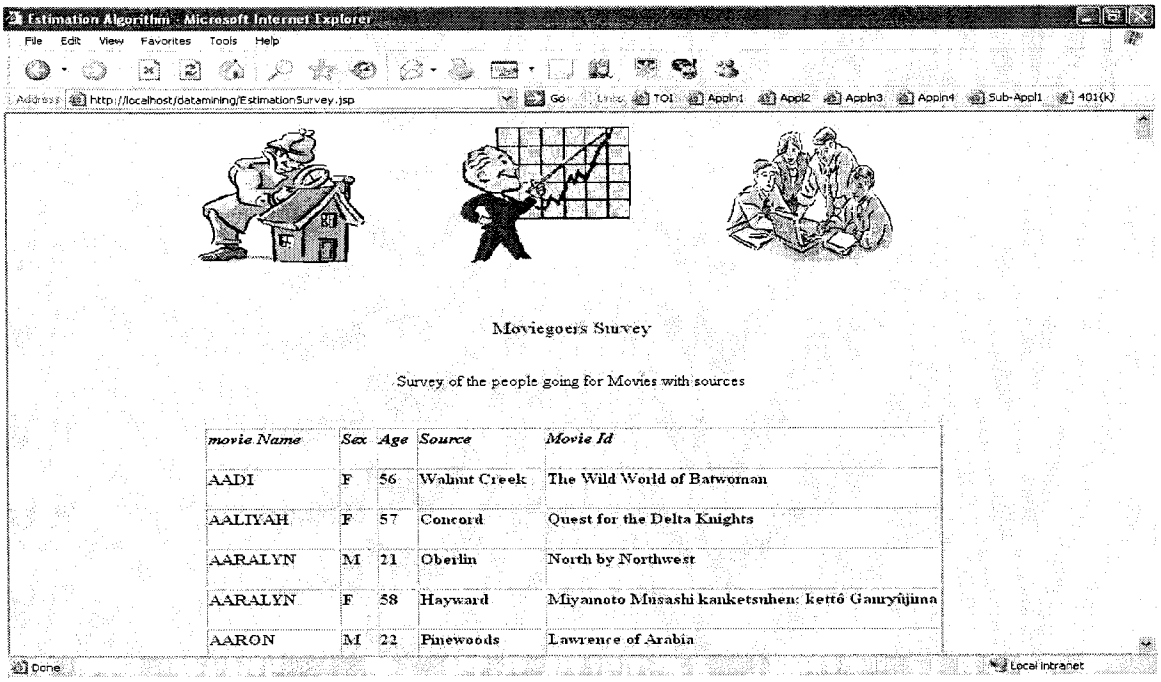


Figure D-21 EstimationSurvey.jsp

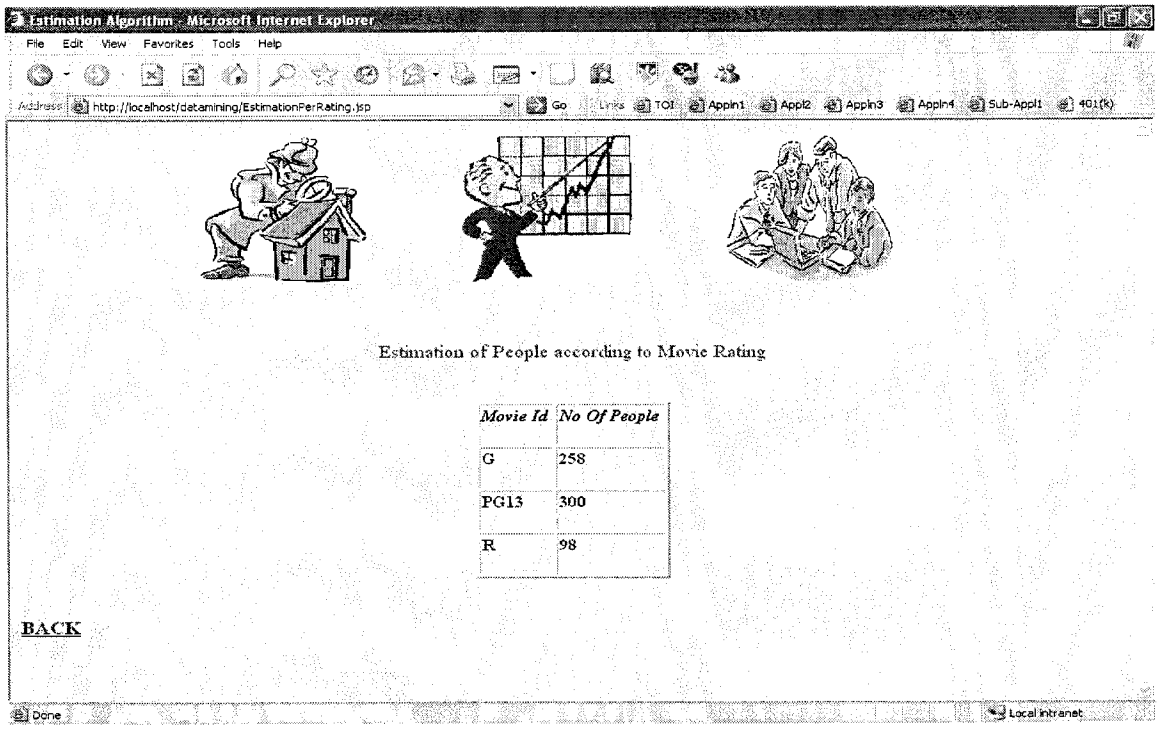
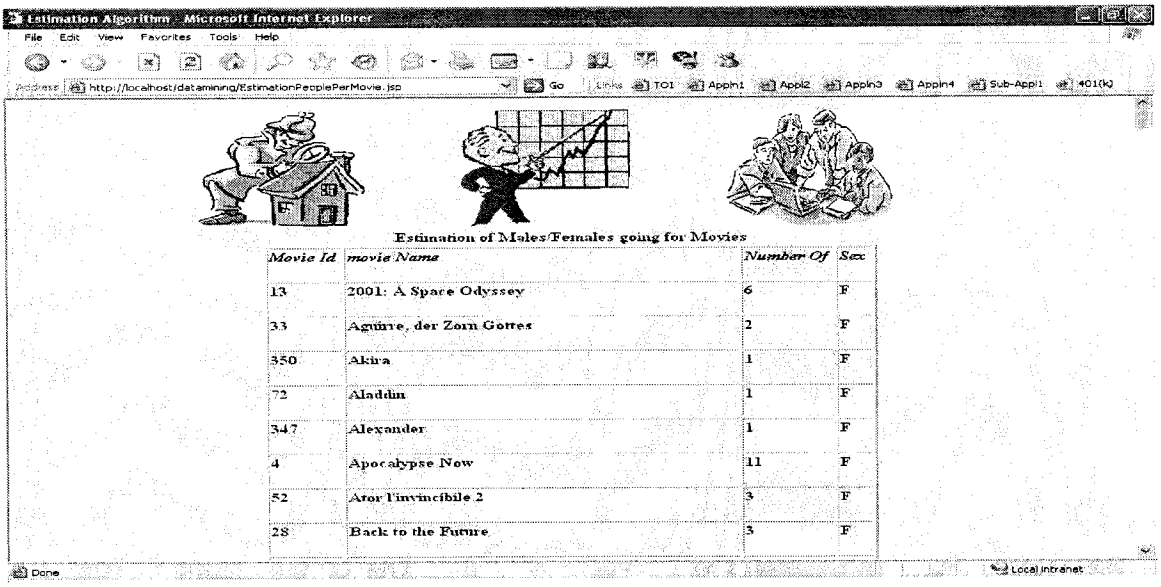


Figure D-22 EstimationPerRating.jsp



Estimation of Males/Females going for Movies

Movie Id	*movie Name*	*Number Of*	*Sex*
13	2001: A Space Odyssey	6	F
33	Aguirre, der Zorn Gottes	2	F
350	Akira	1	F
72	Aladdin	1	F
347	Alexander	1	F
4	Apocalypse Now	11	F
52	Aror l'invincibile 2	3	F
28	Back to the Future	3	F

Figure D-23 EstimationPeoplePerMovie.jsp